InQuira 6 Language Guide

Developing and Maintaining InQuira 6 Dictionaries
Version 6.2

Document Number IQ62-LDG-07

January, 2004

InQuira
851 Traeger Ave
Suite 125
San Bruno, CA 94066

Contents

Preface

About the InQuira 6 Language Guide

This guide provides an overview of InQuira 6 language development. It describes the InQuira 6 language analysis process, the language processing components, the language development environment and tools, and the tasks involved in customizing language processing for an application.

This guide is intended for application developers who want to implement InQuira 6 language processing for an application. Throughout the language development process, you may also need to consult the *InQuira 6 Application Guide* for additional application configuration information. See ***InQuira 6 Documentation*** on page ix for a complete description of the InQuira 6 product documentation.

This preface includes information on:

- the general organization of this guide
- the support services available from InQuira Customer Support
- the available product documentation and its conventions

In This Guide

The *InQuira 6 Language Guide* is divided into the following sections:

Introduction to Language Development **on page 13**

This section provides an overview of InQuira 6 language development, including requirements and prerequisites, the language development environment and components, and the language development process.

InQuira 6 Language Analysis **on page 17**

This section describes InQuira 6 language analysis, the various components and their functions, including analyzing user requests, formulating answer criteria, and determining the best response.

Working with the Dictionary **on page 29**

This section describes the Dictionary, its architecture, contents, and components, and provides information on how the Dictionary components interact, and how you create and modify the language components to customize language processing for your application.

Working with InQuira Match Language **on page 61**

This section provides an overview and detailed syntax description of the IML, which you use to create expressions for matching information in user questions and application content.

Using the Dictionary Manager **on page 105**

Provides information on using the Dictionary Manager, the graphical user interface to all of the functions of the Dictionary.

Customizing the Application Dictionary **on page 133**

This section describes the processes and procedures for updating the Dictionary, including creating, modifying, and testing Dictionary concepts, rules, and alias lists.

Testing the Application Dictionary **on page 153**

This section describes the processes and procedures for testing the effects of updating the Dictionary, including the Test Drive and Regression Testing applications of the Workbench.

Contacting InQuira

You can contact InQuira by mail, telephone, fax, and email.

Mail:	851 Traeger Ave Suite 125 San Bruno, CA 94066
Telephone:	(650) 246-5000 InQuira Customer Support Hotline: (888) 947-8324
Fax:	(650) 246-5036
Email:	For sales information, send email to **sales@inquira.com**. For product support, send email to **support@inquira.com**.

You can find out more about InQuira on the web at: **www.inquira.com**.

Note: See ***InQuira Customer Support*** on page ix for more information on reporting incidents to InQuira Customer Support.

InQuira Customer Support

InQuira Customer Support is available from 6:30 am to 4:30 pm PST, excluding InQuira holidays.

You can contact InQuira Customer Support by email at: **support@inquira.com**. We recommend that you use email to report all Priority 2, 3, and 4 incidents.

For Priority 1 incidents, please use the support hotline: (888) 947-8324.

Important: We accept Priority 1 requests only by telephone. We recommend that you send a follow-up email for Priority 1 requests after contacting InQuira Customer Support using the support hotline.

Call response times are determined by the following priority definitions:

Priority Level	*Response Time*	*Definition*
1	1 business hour	A production system hangs or crashes, or continued use of the program is impossible.
2	8 business hours	The product is usable with major restrictions on functionality.
3	16 business hours	The product is usable with minor restrictions on functionality.
4	3 business days	You have a question or an enhancement request pertaining to the software or the documentation.

InQuira 6 Documentation

InQuira 6 is shipped with the following documentation set. Each document in the set contains a specific type of information to help you use the product.

Document	*Number*	*Description*
InQuira 6 Installation Guide	IQ62-IG-07	This guide is intended for technical staff who are responsible for installing InQuira 6. It provides detailed information on installing and configuring InQuira 6 products and components.

InQuira 6 Application Guide	IQ62-ADG-07	This guide is intended for application developers who need to develop and deploy an InQuira 6 application. It describes InQuira 6 integration, development, configuration, and maintenance.
InQuira 6 Language Guide	IQ62-LDG-07	This guide is intended for application developers and subject matter experts who need to create, implement, and maintain aspects of the InQuira 6 Dictionary. It provides an overview of the Dictionary components, and describes customization, maintenance, and administration tasks and the tools, processes, and procedures required to perform them.
InQuira 6 User Interface Guide	IQ62-UIG-07	This guide is intended for application developers who want to integrate and customize the InQuira 6 Dynamic Portal User Interface. It contains information about the elements and features of the User Interface, and provides guidelines for integrating it into your web architecture, customizing its appearance and functionality, and implementing various special features.
InQuira 6 Analytics Guide	IQ62-AG-07	This guide is intended for application developers who want to configure, deploy, and maintain InQuira 6 Analytics, and for business analysts who want to use InQuira 6 Analytics to report on InQuira 6 performance.

Documentation Delivery

InQuira 6 documentation is distributed with the software release as a set of Portable Document Format (PDF) files. InQuira documentation is available only to licensed users of our software products and may not be redistributed in any form without express permission from InQuira, Inc.

Note: You need the Adobe Acrobat reader to view PDF documents. The Adobe Acrobat reader is available from Adobe Systems at: **http://www.adobe.com**.

If you encounter a problem, need help using the documentation, or want to report an error in the content, please contact InQuira Customer Support, as described in ***InQuira Customer Support*** on page ix.

If you need help obtaining InQuira product documentation, or want to obtain permission to redistribute a portion of the contents, contact your InQuira account representative.

Documentation Conventions

We use the following typographical conventions in our documentation:

Convention	Definition
`monospace`	We use monospace font to denote code examples, path and file names, parameter values, and system messages and prompts.
italics	Italics indicate terms contained in the glossary and citations of other published sources. Strings of italicized text within file names or parameters indicate variable text that you must replace with an appropriate string, or that the system will replace with product- or installation-specific values. Product directories may use italicized *n* characters as variables to denote current product version numbers.
`<value>`	Indicates a symbolic for which you must specify an appropriate value. For example, `<section>` indicates a specified section of the answer display page.
`%`	Indicates the Unix command-shell prompt without root privilege.
`C:\`	Indicates the Microsoft Windows command prompt or file system.
`[item]`	Indicates an optional item within a syntax description.
`{item}`	Indicates a required item within a syntax description, only when necessary for clarity. Otherwise, required items appear without braces.
|	Separates choices in lists of optional and required items within syntax descriptions. For example, `[item1|item2]` or `{item1|item2}`.

`$TERM`
or
`${TERM}`
	Indicates a variable (within Unix-type environments) for which you specify an appropriate value. For example, $ROOT indicates the root directory of the installed product. Variables are sometimes enclosed in curved braces when necessary for clarity.
	
	In some instances, variables are set by and/or resolved by the system with no user intervention required.

`%TERM%`	Indicates a variable (within Microsoft Windows environments) for which you specify an appropriate value.

Screen and Text Representations

The depictions of product screens, screen text, and file contents in our documentation are representations. We attempt to accurately convey the product's appearance and functionality; however, actual product contents and displays may differ from the published examples.

World Wide Web Resources

We provide Uniform Resource Locators (URLs) for various relevant Web resources for your convenience. We attempt to provide accurate information; however, these resources are controlled by their respective owners, and are therefore subject to change at any time.

Chapter 1

Introduction to Language Development

InQuira 6 works by interpreting requests from users, determining what information will satisfy those requests, and presenting the best responses from the content (documents, web pages and other data) that you have configured for use with the application.

InQuira 6 uses a component called the Dictionary to store the information that it uses to determine response criteria for each user request. The Dictionary installed with your application is engineered specifically for the subject area that your application addresses.

The language development process consists primarily of tailoring the application Dictionary to the specific requirements of your business environment. In some cases, the Dictionary installed with your application may already contain application-specific Dictionary information engineered by InQuira staff or by an InQuira partner.

This section provides an overview of the following aspects of InQuira 6 language development:

- the language development process and the tasks associated with that process
- the features of the language development environment
- the components used in language development

The Language Development Process

The language development process consists of configuring, customizing, and testing the application Dictionary to achieve the desired interaction between end-user's questions, the application content, and the presentation elements of the User Interface.

During language development, you customize the Dictionary so that the application recognizes, interprets, and responds to user requests in ways that:

- satisfy user requests
- achieve business ojectives

- optimize the effectiveness of the application content

The parts of the Dictionary that you add or modify during language development include Concepts, Rules, and Alias Lists. See ***Language Development Components*** on page 14 for more information on Dictionary objects.

Language Development Tasks

InQuira 6 language development is the process of supplying application-specific information about your business environment, including the content subject matter and business environment to ensure the best possible responses to user requests. Language development tasks include:

- adding new words and phrases, such as product and model names, and specifying their relationships with one another
- adding ways to match specific types of user requests with the preferred application responses
- tailoring the application's behavior based on user profile and session information (requires the Business Condition module)

Language Development Environment

To perform language development tasks, you must have access to an InQuira 6 language development environment. The language development environment consists of a configured InQuira 6 application, including a configured Dictionary, Content Store, run-time environment, and Workbench, which contains the Dictionary Manager, User Manager, Test Drive, InQuira 6 Analytics Administration, and Regression Tester applications.

See *InQuira 6 Installation Guide* and *InQuira 6 Application Guide* for more information on installing and configuring an InQuira 6 application.

Language Development Components

The Dictionary contains semantic information about words and phrases, their meanings and relations, the intent of user requests, and the actions and responses that are available to the application.

The Dictionary encodes this information within the following types of objects:

- Rules, as described in ***Dictionary Rules*** on page 15

- Concepts, as described in ***Dictionary Concepts*** on page 15
- Alias Lists, as described in ***Dictionary Alias Lists*** on page 16

You access the Dictionary objects through the Dictionary Manager application in the Workbench as described in ***Using the Dictionary Manager*** on page 105.

Dictionary Rules

Rules are Dictionary objects that control how the application recognizes, interprets, and responds to user requests. Each Rule specifies one or more conditions and and consequences, or actions. Conditions are criteria for matching some portion of a user request. Actions are the operations that the application can perform in response to matched conditions.

The Rules in the Dictionary constitute a set of instructions for correctly interpreting and responding to user requests in the context of the your business environment. They provide the framework that the application component called the Rules Engine uses to determine response criteria for a user request. Rules also specify the actions, including various types of information retrieval, that will best satisfy that criteria.

Rules contain various subcomponents, some of which are expressed in a special language, called InQuira Match Language, or IML. See ***Working with Rules*** on page 31 for more information about Rule components. See ***Working with InQuira Match Language*** on page 61 for more information about IML.

You can access, create, and modify the Rules through the Dictionary Manager as described in ***Using the Dictionary Manager*** on page 105.

Dictionary Concepts

Concepts are Dictionary objects that define meanings of words and phrases that occur both in user requests and in the application content.

Concepts have relationships with other concepts. These relationships form a hierarchical linguistic model, called the Ontology. The Ontology is a Dictionary component that is organized into layers that correspond to areas of conceptual knowledge, such as a base language, an industry, or a particular business environment. These discrete areas of knowledge are called Domain. An application Ontology is a pre-packaged collection of multiple layers, arranges so that only relevant conceptual knowledge is included.

You can access, create, and modify concepts directly through the Dictionary Manager without any specialized knowledge of linguistics or the structure of the Ontology. For more information, see ***Adding and Modifying Concepts*** on page 143.

For more information on Concepts and their relationships, and the Ontology, see ***Working with Concepts*** on page 49.

Dictionary Alias Lists

Alias Lists are simple lists that provide a means of substituting one term for another for use in other operations that the application performs.

Alias Lists are collections of these substitution pairs that you can reference as a single logical entity. For example, there is a base domain Alias List that maps numbers to digits, and a financial industry domain Alias List that maps company names to stock symbols.

Base language and industry level Dictionary domains contain several pre-configured Alias Lists that support basic substitution operations.

You can view, create, and modify Alias Lists using the Dictionary Manager. See ***Working with Alias Lists*** on page 52 for more information on Alias Lists.

Chapter 2

InQuira 6 Language Analysis

InQuira 6 performs language analysis during two phases of it's operation: content processing and request processing. Content processing is the means by which InQuira 6 collects, standardizes, and indexes the application content. Request processing is the means by which the application responds to a user's question or request.

During content processing, the the application identifies the concepts that occur in the application content, performs additional semantic analysis, and records the results in various indexes within the application Content Store.

During request processing, the application determines the context, conditions, and concepts that occur in user requests by matching the content of the request against the information in the Dictionary.

Language Analysis Components

The InQuira 6 components that interact to perform language analysis are:

- the Content Store, which stores the application content and associated data
- the Language Analyzer, which performs basic semantic analysis of both user requests and application content
- the Dictionary, which contains the Concepts and Rules that the Rules Engine uses to determine the criteria for the best possible answer
- the Rules Engine, which matches user requests to Rules by comparing request content to business conditions and language conditions, and generates a list of actions
- the Evaluator, which scores and ranks the results of the various actions to determine the best possible answers

The Language Analysis Process on page 18 describes the interaction of the language analysis components during request processing.

The Language Analysis Process

InQuira 6 performs language analysis during both content processing and request processing.

During content processing, InQuira 6 collects, standardizes, and indexes the application content. The Language Analyzer analyzes each word in the content and establishes its relationship to:

- the Concept information defined in the Dictionary
- the structure of the document in which it is located

The Language Analyzer stores the resulting semantic information in the Content Store for use during request processing.

Request processing occurs when an end user submits a question or request to the InQuira 6 application. During request processing, the Language Analyzer:

- analyzes the request using the Concepts stored in the Dictionary

and the Rules Engine:

- evaluates the request using the Rules stored in the Dictionary
- compiles the list of actions, which are potential application responses to the request

The Evaluator:

- evaluates the actions, and ranks the responses based on their scores, as described in ***Scoring and Ranking Responses*** on page 21
- formats the responses for display by the User Interface

The Content Store and Indexes

During content processing, the Language Analyzer analyzes each word in the web pages and documents containing unstructured text that are specified to be included in the application content. The Language Analyzer stores the results of its analysis in the following indexes within the application Content Store:

- the title index stores information related to document titles
- the full-text index stores the content of the documents as tokens and their associated Concepts
- the reference index stores meta-tag information for each document including link text that points to a given document on the website

See the *InQuira 6 Application Guide* for more information on the Content Store, the indexes, and content processing.

Analyzing Requests

When a user submits a question or request to the application, the User Interface passes the request to the Rules Engine. The Rules Engine creates an object, called the sentence object, that contains the user input and additional user profile and session information.

During sentence object creation, the Rules Engine calls the Language Analyzer. The Language Analyzer identifies the basic linguistic and semantic components of the user request. This information is used by the Rules Engine during request evaluation.

The Language Analyzer also resolves the meaning of pronouns and other implicit references to concepts that occurred in previous requests within the session, as described in ***Evaluating Anaphora*** on page 24.

Evaluating Requests

The Rules Engine evaluates the sentence object against the set of Rules contained in the Dictionary. The Rules specify conditions that potentially match various aspects of user requests, such as:

- language conditions as described in ***Language Conditions*** on page 31
- business conditions as described in ***Business Conditions*** on page 33

The evaluation process includes the phases described in:

- ***Evaluating Spelling*** on page 19
- ***Evaluating Business Conditions*** on page 20
- ***Evaluating Language Conditions*** on page 20

Evaluating Spelling

The Rules Engine evaluates the spelling of the words in the sentence object using a packaged spelling checker program. The spelling checker program contains lexicons that you can update as part of customizing your application. For more information on the spelling checker program, see the *InQuira 6 Application Guide*.

Evaluating Business Conditions

Business Conditions are special conditions within Rules that match contextual information about the user or the current session. During request processing, a component called the Preference Service collects contextual user and session information from configured sources, and adds this information to the sentence object.

Note: You must have the Business Conditions module installed to use the Preference Service and business conditions within Rules.

The Rules Engine uses the information added by the Preference Service to evaluate Business Conditions, and matches only Rules whose conditions are met by the information in the request.

The Rules Engine evaluates Rules with matched Business Conditions and Rules that do not specify Business Conditions for language conditions, and eliminates Rules with Business Conditions that do not match.

See ***Working with Business Rules*** on page 55 for more information on the Preference Service and accessing session and user context information.

Evaluating Language Conditions

Language conditions are conditions within Rule that match the content of the user request.

Note: The Rules Engine evaluates language conditions only for Rules having true or no Business Conditions. See ***Evaluating Business Conditions*** on page 20 for more information.

You set language conditions within Rules to specify actions based on the language content of the user request. The content that you can match includes actual words and phrases as well as more general meaning or intent. You can specify language conditions in the form of:

- natural language example questions to match and exclude from matching, as described in ***Question Examples*** on page 32
- question patterns specified in IML, as described in ***Question Patterns*** on page 32

The Language Analyzer performs additional analysis on various semantic elements (words and phrases) as requested by the Rules Engine during this phase.

Defining Response Criteria

The Rules Engine defines response criteria by creating a list of the actions specified by the Rules that are true for the request. Rules can specify one or more of several types of actions, as described in ***Types of Actions*** on page 21.

The list of actions associated with a particular request constitutes the criteria for determining the best responses, as well as criteria for categorizing the responses for display in the Dynamic Portal User Interface.

Types of Actions

Rules can specify both information retrieval actions and actions that affect how the Rules Engine will process the current Rule or subsequent Rules. Information retrieval actions include:

- searching the indexed content in the Content Store
- executing a structured data query
- displaying custom content
- displaying glossary definitions

Rule processing actions include:

- setting an attribute for use in Business Conditions
- restarting the request-response process with a new question

See ***Actions*** on page 33 for more information on the types of information retrieval actions you can specify.

Evaluating Actions

The Evaluator analyzes the actions in the action list by ranking the responses they produce by their response scores. The high-scoring results represent the best available responses to the user request, which are displayed in the appropriate display area of the User Interface.

In applications that implement the InQuira 6 Dynamic Portal user interface, responses are categorized and presented in separate areas (*portlets*) of the response page depending on the answer purpose specified in the Rule that generated the response. See ***Answer Purposes*** on page 35 for more information on specifying answer purposes in Rules.

Scoring and Ranking Responses

InQuira 6 scores and ranks responses from unstructured content in order to determine the best possible response to a user request. A response's score comprises several elements that the application combines and weighs to calculate a final score for each potential response.

The elements of a response's score include:

- the search component score
- the document relevance score
- the document recency score

The Scoring Process on page 22 describes the methods you can use to configure response scoring within the application, and the application functions that are affected by the scoring process, or that use scoring information.

The Scoring Process

Response scoring begins when the Evaluator processes the action list associated with a request. The action list can contain a large number of actions, including Search Components that specify multiple search criteria.

During action evaluation, the Evaluator considers every word in the indexed content as a potential match for the search criteria contained in the action list. A question will generally have multiple responses, and each response that the Evaluator identifies has an associated document excerpt, which is the compiled text that the User Interface displays as an answer on the results page.

To determine response scores for results, the Evaluator:

- matches the search criteria to the words within the indexed content, and generates a score for each associated excerpt, as described in ***The Search Component Score*** on page 23
- evaluates document meta-data associated with each excerpt to determine document recency and document relevance scores, as described in ***The Document Relevance and Recency Scores*** on page 24
- uses a weighted averaging technique to combine these elements into a final score, as described in Setting Relative Weights for Scoring.

Each excerpt that the application generates as a response receives a score. Each response's score is made up of several score values, which the application combines and weighs to calculate the total response score. The response score range is from 0 to 1.0, with 1.0 being the highest possible score.

Scoring Usage

The application uses response scores in several ways:

- the User Interface uses response scores to determine which results are within the display thresholds
- the InQuira 6 Analytics application uses scoring information in the Concept Scoring, Question Scoring, and System Accuracy reports, as described in *InQuira 6 Analytics Guide*.
- the Test Drive application displays scoring information on the Unstructured Results tab, as described in ***Testing the Application Dictionary*** on page 153.

Setting Scoring Values

The algorithms and range for response scoring are not configurable. Response scoring uses a fixed range from 0 to 1.0. However, you can set the relative weights of the subscores that make up the total response score.

The Dictionary contains a Rule called Launch Document Search that sets the following default weights for the search component, relevance, and recency scores:

Score Component	***Default Value***
Search Component	99.999999
Document Relevance	0.000001
Document Recency	0

You can set weight values within individual Rules to override default settings as described in Rule Modifying Actions and ***Search Components*** see "Working with Search Components" on page 44. You can set weight values to apply to a single Rule (locally) or to apply to all Rules (globally).

Note: You can also set thresholds for minimum score and maximum answers for each defined answer purpose in the application configuration, as described in *InQuira 6 Application Guide*.

The Search Component Score

Search criteria are specified by Search Components, which are collections of search criteria and related parameters associated with a word or phrase in the user request. Any number of Search Components can be assigned to a particular request, and a given Search Component can match individual words or a range of words defined by proximity and semantic relationships. Each Search Component has a specified score value, which is an indicator of its value in relation to the condition that it is associated with.

See ***Working with Search Components*** on page 44 for more information.

Each Search Component range is associated with one or more excerpts defined by the Excerpt Service. A given excerpt can contain multiple search components of varying values.

The Evaluator calculates a score for each excerpt based on:

- the scores of the Search Components that the excerpt contains
- the proximity of the Search Components within the excerpt

Excerpts that contain enough high value search components are considered candidate responses. The Evaluator then calculates the document relevance and recency scores for the candidate excerpts.

The Document Relevance and Recency Scores

Each excerpt is associated with a document. During content processing, the Reference index of the Content Store stores information about:

- each document's popularity and relevance within the collection
- the last update to the document

The Evaluator assigns document relevance and recency scores based on this information. The relevance and recency scores are combined and weighed with the Search Component score to determine the total response score. The relative weights for response subscores are determined by global settings in the application configuration, as described in *InQuira 6 Application Guide*, or by settings specified within Rules that are override the global settings for this response.

Evaluating Anaphora

InQuira 6 can process requests in which an important concept that occurred in a previous request is replaced by a substitute word or phrase. The term *anaphora* refers to the relationship between the grammatical substitute (usually a pronoun such as *it*) and the previously occurring concept.

During language analysis, InQuira 6 resolves anaphoric expressions by matching the substitute word or phrase (often a pronoun,) with the correct concept occurring in a previous request, and annotating the request with the concept information.

For example, an inital request:

```
Where can I buy a product_name?
```

might be followed by:

```
Does it come in blue?
```

The Language Analyzer processes anaphoric expressions by:

- creating a list of candidate concepts, or *salience list*, for each user request, as described in ***Creating the Salience List*** on page 25
- determining that a word or phrase refers to a concept that occurred in a previous question as described in ***Identifying Anaphoric Expressions*** on page 25
- matching the referring expression to the correct concept as described in ***Resolving Anaphoric Expressions*** on page 27
- annotating the relevant portion of the request for use by the Rules Engine as described in ***Annotating the Anaphoric Expression*** on page 27

Creating the Salience List

The Language Analyzer creates a salience list for each request that it processes. The salience list contains Concepts that occur within the question that are potentially useful for resolving anaphoric expressions within subsequent requests.

Important: The Language Analyzer will not perform anaphora resolution for the first question in a session, as there is no previous question, and therefore, no salience list.

The Language Analyzer adds Concepts to the salience list if:

- the Concept is part of a noun phrase
- the Concept is not a determiner or a pronoun

Note: Pronoun Concepts are added to the salience list only when the Rules Engine has annotated a Concept from a previous anaphora resolution. This level of annotation enables the Language Analyzer to resolve anaphoric expressions within requests that are twice or more removed from the original Concept occurrence.

- the Concept priority level exceeds a minimum threshold. See ***Concept Priority Levels*** on page 51 for more information on Concept priority levels.

The Language Analyzer partially orders the candidate concepts within the salience list using the following criteria:

- Search Component score. See ***Scoring and Ranking Responses*** on page 21 for more information on search component scoring.
- Concept priority level
- sentence position (Concepts having equal Search Component scores and Concept priority levels are ordered according to their position in the request)
- range length (Concepts associated with a longer range of words within the request are ranked higher)

Note: The primacy of the Search Component score is based on the probability that Concepts belonging to industry and application domains, which will usually have higher Search Component scores, are preferred candidates for anaphora resolution.

The Language Analyzer stores the salience list with the session information so that it is available to the Rules Engine for processing the next request.

Identifying Anaphoric Expressions

The Language Analyzer identifies referring expressions using the InQuira Match Language (see ***Working with InQuira Match Language*** on page 61 for more information on IML). The anaphora-specific IML matches referring expressions, and eliminates non-referring pronoun expressions by testing each word in the question against the following criteria:

Expression	Definition	Examples
it (pronoun)	The word *it*, except when used existentially, as in the phrase *is it true that...* or *what does it mean when...*	`Does it come in blue?`
deictics	Words that are direct indicators. Currently defined deictic words are: • `this` • `that` • `these` • `those`	`Where can I buy this?` `Are there any cheaper than these?`
deictic noun phrases	Noun phrases formed from the defined deictics.	`Do you have this product_type in blue?`

When the Language Analyzer identifies a referring expression, it:

- determines the head of the expression, as described in ***Determining the Head of the Anaphoric Expression*** on page 26
- tests whether the referring expression can be resolved within the current request, as described in ***Testing Referring Expressions Within the Current Request*** on page 26

Determining the Head of the Anaphoric Expression

The Language Analyzer determines the head of the referring expression. For single pronouns, the head of the expression is the pronoun itself. For noun phrases, for example, `the car`, the Language Analyzer determines `car` to be the head.

Testing Referring Expressions Within the Current Request

The Language Analyzer checks whether the referring expression can be resolved within the current sentence by:

- creating a salience list for the current request

- checking that each member of the salience list occurs prior to the referring expression (with no overlap)
- checking that and all semantic constraints apply

If the Language Analyzer locates a valid concept within the current sentence's salience list, it uses that concept to resolve the referring expression, and anaphora resolution is complete. For example, in the following request:

```
In the product_name, does it come with air conditioning standard?
```

the pronoun `it` resolves to the concept associated with `product_name`.

Resolving Anaphoric Expressions

The Language Analyzer resolves anaphoric expressions by checking the previous request's salience list to locate any concepts that fit semantically with the referring expression. It checks each noun phrase candidate concept (such as `the car` or `this car`) in order, for semantic constraints. If the semantic constraints fail, it then checks the next list member.

The Language Analyzer locates concepts for anaphora resolution using concept relations defined in the Ontology. The Language Analyzer uses internally defined methods to locate the appropriate *type of* or *part of* relation for a referring expression. For example, the expression `the car` refers to a specific type of car, `model_name`, that occurred in the previous request; and the expression `the wheel` refers to a part of the car, `model_name`, that occurred in the previous request. See ***Concept Types and Relationships*** see "Concept Part-of-Speech and Property Relationships" on page 145 for more information.

A candidate concept can resolve only one referring expression in a request. The Language Analyzer removes concepts from the list once they are used to resolve an anaphoric expression.

Annotating the Anaphoric Expression

The Language Analyzer completes the anaphora resolution process by applying the `#ANAPHORA` label over the range of the request that contains the original referring expression. See ***Language Analysis Labels*** on page 160 for more information on the types of labels that the Language Analyzer assigns during language analysis.

Chapter 3

Working with the Dictionary

The Dictionary is the repository of the language- and business- related information in the application. It contains both general and customer-specific information that the application uses to respond to user requests. During request processing, the application's Rules Engine compares each request to the information in the Dictionary to determine the best possible response.

The Dictionary is a collection of several types of objects organized into functional packages called Domains. The primary Dictionary objects are Rules, Concepts, and Alias Lists.

A complete and functional Dictionary is shipped and installed with the product, and the appropriate base, industry, and in some cases application-specific Domains are installed during the normal installation process.

The Dictionary is designed to be extended and refined during the implementation and maintenance processes. You extend the Dictionary by:

- creating application-specific domains and domain groups as necessary
- defining application-specific Rules, Concepts, and Alias Lists

You access the Dictionary using the Dictionary Manager application of the Workbench, which is described in ***Using the Dictionary Manager*** on page 105.

Dictionary Domains

Dictionary Domains are discrete collections of Dictionary objects that address specific areas of knowledge required by the application. The installed Dictionary contains multiple base- and industry-level packages that combine to make up a complete application Dictionary. Base and industry-level Domains contain:

- Concepts that encode semantic information about words and their relationships with other words that apply to the subject matter area
- Rules that determine how the application will interpret and respond to questions or requests

- Alias Lists that substitute a defined value for a word in a user request for use in some operation, such as substituting a stock symbol for a company name for use in retrieving stock price data.

Domains range in scope from general to specific. The most general Domains contain extensive vocabulary for native languages, such as English. More specific Domains address various industries, such as telecommunications or financial services.

The most specific Domains address the unique characteristics of your application and business environment, such as product and model names, specific processes, and the implementation of special Dictionary methods and features such as Business Conditions and custom responses.

You can view the Dictionary Domains available to your application using the Dictionary Manager. You can use the Dictionary Manager to create application-specific Domains, and to create logical collections of Domains, called Domain Groups.

See ***Using the Dictionary Manager*** on page 105 for information on accessing the Dictionary Manager.

Application-Specific Domains

Application-specific Domains contain Concepts, Rules, and Alias Lists that apply to the specific characteristics of your business environment.

Concepts in an application-specific domain might include product names and terms related to specific processes within your organization. Application-specific Rules might include instructions for retrieving responses to price-of-product inquiries from a specific database, as well as instructions to retrieve specific documents for billing inquiries. Application-specific Alias Lists provide the means to perform specific transformation, such as displaying definitions from configured online glossaries.

You can create any number of application-specific Domains during the implementation process. You can also create Domain Groups that include any number of application-specific, industry, and base Domains.

You create application-specific Domains and Domain Groups using the Dictionary Manager, as described in Creating Domains and Creating and Modifying Domain Groups.

Domain Groups

Domain Groups are logical groups of Dictionary Domains that you can create within your application. Domain Groups provide a mechanism to combine Dictionary objects that reside in multiple Domains into a single logical Domain for use within your Dictionary.

For example, a financial services application might require a Domain Group that includes a base English Domain, a financial services industry Domain, and one or more application-specific Domains.

You create Domain Groups using the Dictionary Manager, as described in Creating and Modifying Domain Groups.

Working with Rules

Rules are Dictionary objects that control how the application recognizes, interprets, and responds to user requests. Each Rule specifies one or more *conditions* and one or more resulting consequences, or *actions*. Rules also have special characteristics, such as precedence and scoring, that the Rules Engine uses during request processing.

The Rules in the Dictionary constitute a set of instructions for interpreting and responding to user requests in the context of your business environment. An ideal set of Rules for an application would form a complete representation of possible user requests and instructions for the preferred responses to those requests.

You create application-specific Rules by specifying appropriate conditions and actions to capture the important user requests and respond with the preferred information from your application's content.

You can specify the following types of conditions within Rules:

- ***Language Conditions*** on page 31
- ***Business Conditions*** on page 33

The actions that you can specify within Rules are described in ***Actions*** on page 33.

Language Conditions

Rules can specify conditions that set matching criteria for the words used within a request. You can specify either of the following types of language conditions:

- question examples, which are expressed as lists of example questions to match and exclude from matching, as described in ***Question Examples*** on page 32
- question patterns, which are expressed in InQuira Match Language (IML), as described in ***Question Patterns*** on page 32

See ***Working with InQuira Match Language*** on page 61 for information on IML syntax and use.

Question Examples

Question examples are natural language phrases, sentences, and questions that you specify as matching criteria within Rules. You specify question examples in plain language, as they would be entered by users. You can specify a question example as either:

- a question that should match
- a question that should not match

to define a range of valid questions for the Rule.

You can associate question example matches with any of the available Actions, as described in ***Actions*** on page 33. During request processing, the Rules Engine uses statistical methods to evaluate user requests against question examples.

You specify question examples within Rules by entering one or more questions to define matching criteria, as described in ***Specifying Question Examples*** on page 135.

Question Patterns

Question patterns are symbolic expressions that you specify as matching criteria within Rules. You use question patterns to match the variety of words and phrases that can be used to express a particular request for information or assistance.

You compose question patterns using InQuira Match Language (IML), which is a regular expression language designed to capture the variations in written language that can be used to express requests.

An IML expression in a Rule condition is an abstract representation of a set of meanings, or an *intent*, that can be expressed by various combinations of actual words or phrases. For example, in an e-commerce environment, *cost of product* is an important intent to represent as a question pattern.

Question patterns can range from simple, general expressions, to very complex and specific ones. The packaged Dictionaries contain basic and industry-level Rules that work together to produce high-quality results for many applications with a minimal layer of application-specific Rules and Concepts.

You specify question patterns within Rules by entering a valid IML expression to set matching criteria, as described in ***Specifying Question Patterns*** on page 135. You can associate question pattern matches with any of the available Actions, as described in ***Actions*** on page 33.

See ***Working with InQuira Match Language*** on page 61 for more information on creating IML expressions.

Business Conditions

Business Conditions are expressions within Rules that specify conditions based on contextual information about the user or the session. The contextual information is provided by the InQuira 6 Preference Service, which collects data from configured sources, or *providers* at runtime. Providers can include the InQuira 6 application itself, web and application servers, and external repositories, such as customer relationship management (CRM) applications.

Business Conditions and language conditions together enable you to tailor application responses based not only on the content of the request, but also based on *who* the requestor is, and *when* and *where* the request is being made.

You can specify Business Conditions based on:

- user or customer profile characteristics, such as user name or type of account
- session characteristics, such as date and time or pages visited
- contextual information contained in the text of the request, such as reference to a specific product name or process

Important: You must have the InQuira 6 Business Conditions module licensed, installed, and configured to use the Preference Service and Business Conditions.

During request processing, the Rules Engine compares the Business Conditions specified within Rules to the values provided by the Preference Service for a given request. See the *InQuira 6 Application Guide* for more information on configuring the Preference Service to access business data for your application.

You specify Business Conditions by:

- adding a Business Conditions section to a Rule
- specifying one or more condition expressions using the available Preferences and Providers configured for your application

as described in ***Working with Business Rules*** on page 55.

Actions

Actions are components of Rules that specify an operation for the application. When the Rules Engine evaluates an Rule as true, the applicaton performs the specified Action or Actions. You can specify multiple Actions within Rules. The Rules Engine processes Actions for all Rules that are true during request processing.

You specify Actions in the Actions section of the Rules window in the Dictionary Manager. You can specify the following types of Actions within Rules:

- search the indexed content, query a configured data source, and display a managed response, such as custom content, as described in ***Answer Actions***
- add Search Components to the request as described in ***Working with Search Components*** on page 44
- restrict the scope of the search within the indexed content as described in ***Content Restriction*** on page 42
- alter the scoring process, as described in ***Relative Weights*** on page 41
- suppress processing of one or more subsequent Rules as described in ***Rule Suppression*** on page 42
- set the maximum number of responses as described in ***Set Result Maximum*** on page 43
- set an attribute for use by a Business Condition as described in ***Setting Context Variables for Business Conditions*** on page 43

Answer Actions

Answer Actions are the means of specifying the methods that the application will use to generate responses. Answer Actions include searching the indexed content, querying a configured data source, and managed responses, such as displaying custom content. You can specify the following types of Answer Actions:

- display custom content as described in ***The Custom Content Answer Method*** on page 37
- match a specific excerpt within the indexed content as described in ***The Specific Excerpt Answer Method*** on page 38
- search the indexed content as described in ***The Search Answer Method*** on page 38
- retrieve structured data as described in ***The Structured Query Answer Method*** on page 39
- display a glossary definition as described in ***The Glossary Answer Method*** on page 40
- display a response using an external (plug-in) method, as implemented in the anaphora evaluation process described in ***Evaluating Anaphora*** on page 24

You can specify multiple Answer Actions of various types within a single Rule, and you can specify additional Actions within a Rule as described in ***Actions*** on page 33.

Answer Actions have the following associated parameters:

- ***Answer Purposes*** on page 35

- ***Answer Methods*** on page 36
- ***Answer Priorities*** on page 41

Answer Purposes

Answer Purposes are categories to which you assign Answer Actions within Rules that correspond to display characteristics defined in the User Interface.

Answer Purposes enable you to establish consistent, focused, and targeted presentation for various types of application content, such as general site information, online glossaries, promotional material, and site features, such as calculators and other tools.

InQuira 6 is shipped with a standard set of Answer Purposes, described in ***Default Answer Purposes*** on page 35, that are designed for use with the Dynamic Portal User Interface. The default Answer Purposes associate each Answer Purpose with a defined response category area, or *portlet*, of the answer page.

You use Answer Purposes by:

- assigning Answer Purposes to Actions within Rules, as described in ***Specifying Answer Purposes*** on page 138.
- configuring portlet presentation in the User Interface, as described in Configuring Answer Purposes in the *InQuira 6 User Interface Guide*.

Note: In contrast with Answer Purposes, Answer Methods correspond to type of data or method used to supply the answer. Examples of Answer Methods include querying structured data, searching the indexed unstructured content, and displaying custom content. See ***Answer Methods*** on page 36 for more information on Answer Methods.

Default Answer Purposes

The standard set of Answer Purposes described below are designed for use with the Dynamic Portal User Interface.

Purpose	Description	Default Presentation
Answer	Content returned that directly addresses the user's question.	Answer area of the response page
Act	Links that provide actions that the user can take on the web site.	Act Now portlet
Promote	Cross-sell or up-sell advertisements for products related to the intent of the question.	Promotion portlet

Link to Category	Links to major topic categories defined for the web site.	Learn More portlet
Define	Links to terms used in the question as well as similar content.	Definition portlet
Jump to Page	Content configured in the Dictionary for use with the direct page display feature.	See Implementing Direct Page Display in the *InQuira 6 User Interface Guide.*
Converse	Conversational response intended for use with a virtual representative on the response page.	See Implementing a Virtual Representative in the *InQuira 6 User Interface Guide.*
Featured Content	Specific featured content from the web site that supplements the answers for a given intent.	Featured Content area of the response page

Answer Methods

Answer methods specify how the application will execute the action specified within the Rule. Answer methods are the means of specifying:

- the various types of Answer Actions as described in ***Answer Actions***
- question context variables for use in Business Conditions as described in ***Setting Context Variables for Business Conditions*** on page 43

Use this Method...	*To...*
Custom Content	create the response using the associated custom content as described in ***The Custom Content Answer Method*** on page 37
Specific Excerpt	create the response using the specified excerpt as described in ***The Specific Excerpt Answer Method*** on page 38
Search	create the response from the results of a search of the indexed unstructured content as described in ***The Search Answer Method*** on page 38
Structured Query	create the response from the results of a query against a structured data source as described in ***The Structured Query Answer Method*** on page 39
Glossary	create the response from the glossary entry associated with the specified term as described in ***The Glossary Answer Method*** on page 40

Plug-In	create the response using an external method. InQuira 6 includes external methods for implementing question follow-up and anaphora resolution as described in ***Evaluating Anaphora*** on page 24.
Set Attribute	set a context variable that the application will set as the action for this Rule, as described in ***Setting Context Variables for Business Conditions*** on page 43

The Custom Content Answer Method

You can specify to display custom content using the Custom Content method in the Answer Actions section of a Rule. Custom content can include:

- custom answers that you create by specifying a title, URL, and answer text within the Answer Action tab
- answers that implement the direct page display feature, as described in xref

You specify custom content answers using the following fields:

Field	*Description*
Title	Enter text to display as the answer title in the User Interface.
URL	Enter the URL that the answer will link to. The User Interface will create the link using the answer title as the link text.
Custom Content	Enter text or valid HTML to display as the content of the answer. **Important:** To use this method to specify direct page display, (Jump-to-Page Answer Purpose), enter the URL in this field, and not in the URL field. See xref for more information.

You can use the Custom Content method in conjunction with:

- any Answer Purpose as described in ***Answer Purposes*** on page 35
- any Answer Priority as described in ***Answer Priorities*** on page 41

You can also use the Custom Content method in conjunction with the following advanced Actions:

- ***Content Restriction*** on page 42
- ***Relative Weights*** on page 41
- ***Rule Suppression*** on page 42

- ***Set Result Maximum*** on page 43
- ***Setting Context Variables for Business Conditions*** on page 43

The Specific Excerpt Answer Method

You can specify to display specific excerpts within the Content Store that you have identified as the preferred answer using the Specific Excerpt method in the Answer Actions section of a Rule.

You specify the following parameters to define the excerpt or excerpts that you want to include as responses:

Field	*Description*
Custom Link Text	Enter the text to use as the answer title.
Title	Enter text to match within excerpt titles.
Specific Excerpt	Enter text to match within excerpts.

You can use the Specific Excerpt method in conjunction with:

- any Answer Purpose
- any Answer Priority

You can also use the Specific Excerpt method in conjunction with the following advanced Actions:

- ***Content Restriction*** on page 42
- ***Relative Weights*** on page 41
- ***Rule Suppression*** on page 42
- ***Set Result Maximum*** on page 43
- ***Setting Context Variables for Business Conditions*** on page 43

The Search Answer Method

You can specify to search within indexed unstructured content using the Search method in the Answer Actions section of a Rule. You specify the following parameters to determine search characteristics:

Field	Description
Retrieve document excerpts using this IML	Enter any valid IML to specify matching criteria for the excerpts generated by the Rules Engine. You can use this field independently of, or in conjuction with, the default ranking option.
Retrieve document excerpts using default ranking	Select this option to return results using the standard Search Component assignment and scoring process. You can use this field independently of, or in conjuction with, specific IML to match document excerpts.

You can use the Search method in conjunction with:

- any Answer Purpose
- any Answer Priority

You can also use the Search method in conjunction with the following advanced Actions:

- ***Content Restriction*** on page 42
- ***Relative Weights*** on page 41
- ***Rule Suppression*** on page 42
- ***Set Result Maximum*** on page 43
- ***Setting Context Variables for Business Conditions*** on page 43

The Structured Query Answer Method

You can specify retrieval from configured sources of structured data using the Structured Query method in the Answer Actions section of a Rule. You specify the following parameters to determine retrieval characteristics:

Field	Description
Data Source	Select a configured structured data source to query. See the *InQuira 6 Application Guide* for information on configuring data sources for structured information retrieval.
Title of Results	Enter plain text to specify a title under which the query results will be displayed.
Description	Enter plain text to specify an optional description for the query results.
Structured Query	Enter the SQL query to submit.

IML to restrict Structured Query	Enter optional IML to restrict the results of the query. The Rules Engine will filter the query results based on matching the specified IML. See ***Working with InQuira Match Language*** on page 61 for more information.
Table rows to display with description	Enter the number of rows to display per result.

You can use the Structured Query method in conjunction with:

- any Answer Purpose
- any Answer Priority

You can also use the Structured Query method in conjunction with the following advanced Actions:

- ***Rule Suppression*** on page 42
- ***Setting Context Variables for Business Conditions*** on page 43

The Glossary Answer Method

You can specify to display terms and definitions from prepared glossaries entry using the Glossary method in the Answer Actions section of a Rule. The Glossary method accesses a special Alias List, named `glossary`, as described in ***Accessing Glossary Information*** on page 53.

Field	Description
Retrieve a glossary entry for the term referenced by this variable	Enter the variable set in the Rule condition that corresponds to the desired glossary entry.

You can use the Glossary method in conjunction with:

- any Answer Purpose
- any Answer Priority

You can also use the Search method in conjunction with the following advanced Actions:

- ***Content Restriction*** on page 42
- ***Relative Weights*** on page 41

- ***Rule Suppression*** on page 42
- ***Set Result Maximum*** on page 43
- ***Setting Context Variables for Business Conditions*** on page 43

Answer Priorities

Answer priorities specify the scoring level that the application will associate with the results of the action. Answer priorities apply to responses that do not use search component scoring, such as Search actions that specify direct IML. The available answer priorities are:

- Highest
- High
- Medium
- Low
- Lowest

See ***Scoring and Ranking Responses*** on page 21 for more information on answer priorities and how they are used within the scoring process.

Relative Weights

You can specify custom values for weighting the score components used to rank unstructured search results using the Add Relative Weights settings option in the Advanced menu of the Rule window. The Rules Engine uses relative weights to calculate the total score for a response during scoring process, as described in ***Scoring and Ranking Responses*** on page 21. The Relative Weights tab provides fields to set the relative weights of:

- the search component score
- the document relevance score
- the document recency score

within the total response score.

Note: The combined value of the relative weights must always equal 100%.

You can apply relative weights to the current Rule (local), or to all Rules that specify the Search method (global).

You specify local relative weights by including a Relative Weights action and an Answer Action using the Search method in the same Rule. The relative weights parameters will apply only to the Search Components that are assigned by the associated Search method.

You specify global relative weights by specifying the Relative Weights action without an accompanying Search Action. The Relative Weights parameters will apply to all assigned Search Components, unless they have specific relative weight settings.

You can specify Relative Weights based on any language conditions and available Business Conditions.

You specify Relative Weights using the Add Relative Weights option, as described in xref.

Content Restriction

You can restrict the scope of Search Actions using the Content Restriction option of the Advanced menu of the Rule window. The Content Restriction tab displays fields to select Collections or Attributes to include in Search Actions. You restrict the Search Action to:

- one or more specified document collections
- one or more specified document attributes

If you restrict searching to multiple document attributes, you must specify an AND or OR relationship between the attributes.

Note: Document collections are defined as part of content processing, as described in the *InQuira 6 Application Guide*.

You can apply content restriction to Search Actions within the current Rule (local), or to all Rules that specify the Search method (global).

You specify local content restriction by including a Content Restriction action and an Answer Action using the Search method in the same Rule. The Content Restriction parameters will apply only to the Search Components that are assigned by the associated Search method.

You specify global content restriction by specifying the Content Restriction action without an accompanying Search Action. The Content Restriction parameters will apply to all assigned Search Components, unless they have specific content restriction parameters.

Note: The last valid Rule that specifies global Content Restriction will override previous global Content Restriction actions, but will have no effect on local Content Restriciton actions.

You can specify Content Restriction based on any language conditions and available Business Conditions.

You specify content restriction using the Add Content Restriction option, as described in xref.

Rule Suppression

You can specify suppression of subsequent Rules using the Suppress Rules option in the Advanced menu of the Rule window. The Rule Precedence window provides options to:

- suppress processing of all subsequent Rules
- suppress processing for only the specified Rules

You can add Rules to the suppression list by dragging from any Dictionary Manager list of Rule names. You can delete Rules from the suppression list by right-clicking and selecting Remove from List.

Set Result Maximum

You can specify the maximum number of results to return for Answer Actions using the Set Result Maximum option in the Advanced menu of the Rule window. The number of results that you select will apply to all of the Answer Actions specified within the Rule. The range of the Results Maximum setting is:

- default, which is set on the Index Service page of the Configuration and Administration Interface
- 1-9
- 10
- 15
- 20
- 30
- 40
- 50

Setting Context Variables for Business Conditions

You can specify Business Condition context variables for use within subsequent Rules in the using the Set Attribute method in the Answer Actions section of a Rule. For example, you could set an attribute based on the occurrence of a phrase such as `my product_name` in a request.

Note: You must have the InQuira 6 Business Conditions module installed to use the Set Attribute method.

The Set Attribute method provides options for you to:

- select a Business Condition from the dropdown list of configured Business Condition

- specify a value to set for the selected Business Condition for use in subsequent Rule processing

as described in ***Working with Business Rules*** on page 55.

Working with Search Components

Search Components are components of Rules that define search criteria based on matching a portion, or *range* of a user request. Assigning a Search Component is one of the Actions available within Rules. Search Components are the primary method of specifying information retrieval for both unstructured (Search method) and structured (Structured Query method) data.

Within the Dictionary, some Rules specify Search Components, while others use Search Components that have been previously assigned by the Rules Engine. The base and industry Dictionary domains contain Rules that specify default Search Components for an application; however you can specify custom Search Components within application-specific Rules to tailor responses to the specific needs of your organization.

A Search Component defines a set of search criteria consisting of one or more IML or SQL expressions. The expressions are ranked within the Search Component by quality, and the Search Component itself is ranked in relation to other Search Components by importance.

A single user request can match multiple Rules, each containing multiple Search Components of varying importance that specify multiple search criteria of descending quality.

You define a Search Component by specifying:

- the variable that associates the matching portion of the user request with the Search Component
- the type of content and the mode of searching
- IML or SQL expressions that define the search criteria
- additional parameters that determine:
    - how the Search Component will interact with other Search Components assigned to the user request
    - how the search results will be scored and ranked within the results set

You define search component elements within the Search Component section of the Dictionary Manager Rule window as described in ***Specifying Search Components*** on page 139.

Search Component Ranges

The range of a Search Component is the portion of the user request that the Search Component matches. Search Components are associated with IML expressions within Rule conditions. When a portion of a request matches an IML expression that assigns a Search Component, the Rules Engine assigns the Search Component to that portion of the user request. A Search Component's range can be as small as a single word, or can contain multiple words and phrases that occur within some specified proximity defined by the IML expression.

Search Component Variables

Variables are the mechanism that associates an IML expression in the condition of a Rule with a Search Component. The IML expression references the variable that designates the Search Component in the form:

```
(IML_expression)=A
```

When a portion of a user request matches the expression (`IML_expression`), the Rules Engine assigns the variable `A`, which is defined within the Rule as a Search Component. The variable is the means of associating the Search Component, and all of its specified criteria, with the range of the user request. The portion of the user request that matches the IML expression constitutes the range of the Search Component.

See ***Variables*** on page 82 for more information.

Search Component Attributes

Search Components have attributes that determine the content source, the mode of retrieval, the ordering of the Search Component within the list of actions, and for structured information retrieval, the various parts of the SQL statement generated by the Rules Engine.

Search Component attributes include types, as described in ***Search Component Types*** on page 45, and associated subtypes, as described in ***Search Component Subtypes*** on page 46.

Search Component Types

The Search Component type is an attribute that specifies the content source and the mode of searching. You can specify the following types of Search Components within Rules:

Index

Index Search Components use IML expressions to define criteria for searching the indexes of unsructured content in the Content Store.

SQL

SQL Search Components use SQL queries to define criteria for retrieving information from configured structured data sources.

Each Search Component type is associated with additional subtype attributes, as described in ***Search Component Subtypes*** on page 46

You specify one or more Search Component expressions in order of priority, or desireability of the results. Each subsequent Search Component expression in the list is assigned a lower quality score, as described in ***Quality Levels*** on page 49.

Search Component Subtypes

Search Component subtypes are attributes associated with the Search Component type attributes described in ***Search Component Types*** on page 45. The Rules Engine uses the type and subtype information when it assigns a Search Component to a request.

For index Search Components, the Rules Engine uses the subtype value to order the assigned Search Components within the action list. The index subtypes currently defined in the Dictionary are described in the following table:

Index Search Component Subtypes	***Description***
standard	Standard is the default for index Search Components. The Rules Engine orders the standard subtypes first within the action list.
ranking	The Rules Engine orders the ranking subtypes after the standard subtypes within the action list.
unspecified	The Rules Engine does not order unspecified subtypes.

For SQL Search Components, the Rules Engine uses the subtype value to populate portions of the SQL statement that it generates to query the data source. For example, an the subtype order would contain the part of the SQL statement that requires results in a certain order, such as like highest to lowest.

The SQL subtypes currently defined in the Dictionary are:

- condition
- field
- binary operator
- table
- order
- limit

- unspecified

When you define a Search Component using the Dictionary Manager, the subtype field displays all the subtypes that are currently used in the Dictionary in the drop-down menu.

Search Component Assignment

During request processing, the Rules Engine assigns all of the Search Components from all matching Rules to a user request. The ranges of the assigned Search Components can can overlap or encompass one another. Search Components having ranges that overlap affect one another according to their Subsume and supplement settings, as described in ***Subsume Settings*** on page 47 and ***Supplement Settings*** on page 48.

Subsume Settings

The Subsume setting specifies whether a Search Component will allow other Search Components that are contained in the same range of the request to be evaluated in the scoring process.

The range associated with a Search Component can contain or overlap ranges associated with other Search Components. The Subsume option specifies whether a Search Component will allow the Rules Engine to process other Search Components that are fully contained within its range. If Subsume is active, the Search Component will not allow the contained Search Components to be used.

Subsume applies only to Search Components having ranges that are fully contained within other ranges. Subsume does not affect Search Components whose ranges simply overlap.

Note: If you define a variable within a Search Component that invokes an addtional Search Component, the Search Component invoked by the variable will be active even if the Search Component that invoked it is subsumed.

Subsuming a Contained Search Component

As an example of using the Subsume setting, consider the following phrase that might appear in a user request:

```
stock options
```

and an IML expression matching the word `stock`:

```
(IML_expression)=A
```

The variable `A` corresponds to a Search Component that specifies a search for a concept `stock` and its synonyms, such as `equity` and `shares`.

Now consider that the word `options` is associated with a different Search Component, associated with the variable `B`, that specifies a search for a concept `options`, and its synonyms. The result is two words, and two assigned Search Components specifying searches for two concepts.

Finally, consider that the two words together, `stock options`, compose a phrase that is associated with a third Search Component, `C`. This Search Component defines search criteria based on the meaning of the entire phrase, `stock options`.

In this example, the phrase `stock options` is a more valuable concept than the parts `stock` and `options` that compose it.

To specify that the Search Component that defines `stock options` search criteria should override the Search Components associated with its constituent words, specify the Subsume option for Search Component `C`.

Specifying Not to Subsume Contained Search Components

In other cases, you may choose not to subsume contained Search Components. Consider the following phrase that might appear in a user request:

```
common stock
```

and an IML expression matching the word `stock`:

```
(IML_expression)=A
```

The variable `A` corresponds to a Search Component that specifies a search for a concept `stock` and its synonyms, such as `equity` and `shares`.

The two words together, `common stock`, compose a phrase that might be also be associated with a Search Component. This Search Component defines search criteria based on the meaning of the entire phrase, `common stock`.

In this example, the whole (`common stock`) is prefered to the part (`stock`); however, results from search for `stock` alone may also be relevant. Therefore, we can specify to preserve, rather than subsume, the contained Search Component.

Supplement Settings

The Supplement setting specifies whether to assign a Search Component if another Search Component has been previously assigned over the same range.

The range associated with a Search Component can contain other ranges associated with other Search Components. The Supplement option specifies whether to add the Search Component if another one is already applied within its range.

If Supplement is set, the Search Component will always be added.

If it is not set, the Rules Engine will check whether there is currently a Search Component applied to any word within the Search Component's range. If so, the Search Component will not be added.

Importance Setting

Importance settings specify the importance of the search component relative to other search components that the Rules Engine analyzes and evaluates.

The importance setting sets a score value range for the set of criteria specified within the search component. Within the score value range, individual search criteria are ranked using the quality setting.

Importance Setting	***Range***
Required	Descends from 100000
Very Important	Descends from 10000
Important	Descends from 1000
Minor	Descends from 100
Trivial	Descends from 10

Quality Levels

You can enter multiple IML or SQL expressions within a search component. You specify the expressions in order of preference, or quality. Quality levels specify the ranking of various search criteria expressions within a single search component.

Search Components can contain multiple expressions in order of priority, which enables the Rule to specify a set of prioritized search instructions in response to matching a portion of the user request. The first level indicates the ideal representation of the concept in the content. The subsequent levels represent alternative representations in order of descending desirability.

Working with Concepts

Concepts are Dictionary objects that define meanings of words and phrases that occur both in user requests and in the application content. Concepts are unique, symbolic, abstract representations of meaning; a single Concept can represent multiple words and phrases that have the same meaning in the context of your application and business environment.

Concepts also have relationships with other Concepts. These relationships specify, for example, other Concepts of which a Concept is a part, or a type. See ***Concept and Cluster***

Relationships see "Concept Relations" on page 51 for more information on Concept relationships.

The various concepts and their specified relationships form a hierarchy of language information that ranges from the base language through industry- and application-specific terminology.

Concepts provide a means for InQuira 6 to:

- treat multiple words and phrases having the same meaning as instances of a single entity
- manipulate, by expanding, contracting, or associating the meanings of words and phrases based on their relations with other Concepts

You use Concepts to define the meanings and relationships of words and phrases that occur in user requests and application content. Customizing Concepts in an application Dictionary consists of:

- adding new application-specific concepts, such as product names
- modifying existing concept definitions to more accurately reflect specific usage in your environment

You create and modify Concepts using the Dictionary Manager, as described in ***Adding and Modifying Concepts*** on page 143.

The Ontology

Concept information within the Dictionary is stored in a special repository called an Ontology. The Ontology is organized into layers that reflect areas of conceptual knowledge relevant to a discrete area, such as:

- a base language, such as English
- an industry or subject matter area
- a specific company or application

The Ontology contains Concepts, and specifies relationships between Concepts that form a hierarchy. ***Concept and Cluster Relationships*** see "Concept Relations" on page 51 describes Concept relationships.

An application Ontology is a pre-packaged collection that comprises multiple layers, so that only relevant conceptual knowledge is packaged for an application.

You cannot view or manipulate the Ontology directly; however, the Concept Tree window of the Dictionary Manager displays a graphical representation of the Concept hierarchy. The Concept Tree window displays the Concept hierarchy from a pre-defined starting point. You can select individual Concepts from the hierarchy display to view detailed Concept information in the Concept window.

Concept Names

Concepts are represented by a unique concept name, which is expressed in the form:

```
part_of_speech.domain_name:concept_headword
```

See ***Specifying General Concept Information*** on page 143 for information on specifying Concept names.

Concept Priority Levels

Concept priority levels set priority for Concepts within the Dictionary. Priority levels influence the scoring of results for request processing, and are also used by the Language Analyzer during anaphora resolution. See ***Scoring and Ranking Responses*** on page 21 for more information about scoring responses, and ***Evaluating Anaphora*** on page 24 for more information about anaphora resolution.

You can set the following Concept priority levels:

- High
- Medium
- Low

In general, use High priority to indicate concepts that are application-specific, or are very important to the application, Medium priority to indicate concepts that are associated with the industry domain, and Low priority to indicate general concepts within the base language.

Concept Synonyms

A single Concept represents multiple words and phrases that have the same meaning. These words and phrases are defined as the Concept's synonyms.

You enter synonyms in the form in which they occur in user requests and application content. You specify a Concept's synonyms using the Dictionary Manager as described in ***Specifying Synonyms for a Concept*** see "Specifying Concept Synonyms" on page 144.

Concept Relations

In addition to the synonyms that define them, Concepts have relationships with other Concepts in the Dictionary, as follows:

Concept Relationships

Concept relationships specify that a Concept posseses, or is possessed as, a property of another Concept. Properties include *type* and *part*, such that a Concept can defined as a *type of*, or *part of* another Concept. See ***Specifying Concept Relationships*** on page 144 for more information on Concept relationships.

Cluster Relationships

Cluster relations specify a similarity in meaning (rather than equilvalence, as in synonym relationships, or possession, as in Concept relationships). See ***Specifying Cluster Relationships*** on page 146 for more information.

Working with Alias Lists

Alias Lists lists are collections of one or more entries that directly map words, phrases, concepts, or global variables to a subtitute value.

Alias List entries are transformative (unlike Concept synonyms). When a word in a user request matches an alias term, the associated alias value is used in place of the original word, usually to enable a specific operation, such as mapping company names to stock symbols.

Glossary Lists are special Alias Lists that you can create to enable the application to access information from online glossaries, as described in ***Accessing Glossary Information*** on page 53.

You add and modify Alias Lists using the Dictionary Manager, as described in ***Adding and Modifying Alias Lists*** on page 147.

Alias Entry Format

The format of an alias entry is:

```
key_term alias_term
```

where:

key_term	is a word or phrase, Concept, or global variable, as shown in the following format examples:
	`word alias_term`
	`phrase_words alias_term`
	`<pos.domain.headword> alias_term`
	`#global_variable alias_term`

alias_term	is the word or phrase that the `key_term` will be transformed into.

See ***Working with Concepts*** on page 49 for more information on Concept formats. See ***Variables*** on page 82 for more information on defining global variables.

Alias List Format

Each alias entry must be entered on a single line. Within the entry, the alias term follows the key term. Key terms and alias terms are separated by a single space (or by a tab within a text file).

The format of an Alias List is:

```
key_term_a alias_term_a
key_term_b alias_term_b
.
.
.
key_term_n alias_term_n
```

Each Alias List entry must begin a new line.

Alias List Names

An Alias List name is an alphanumeric string of any length that identifies the list for reference by the Rules Engine. You cannot use the same name for multiple Alias Lists within a single domain.

You can use the same name for multiple Alias Lists in separate domains; however, if you use two domains that contain lists of the same name within a single application, you should be aware of override conditions as described in ***Adding and Modifying Alias Lists*** on page 147.

Accessing Glossary Information

You can create Glossary Lists, which are special Alias Lists that provide access to glossary information. You add glossary information to the application by creating a standard Alias List and populating it with glossary entries in the format described in ***Glossary Entry Format*** on page 54.

The Rules Engine contains pre-defined Rules to match concepts to key terms in an Alias List named `glossary` by default. You can use these default Rules by:

- defining an Alias List named glossary
- populating the list with information from your organization's glossary source in the glossary entry format

The Rules that access glossary information specify the Glossary Answer Purpose, and are displayed by default in the Definitions portlet of the User Interface.

See ***Adding and Modifying Alias Lists*** on page 147 for more information about working with Alias Lists. See the *InQuira 6 User Interface Guide* for more information about Answer Purposes.

Glossary Entry Format

The format of a Glossary List entry is:

```
key_term [display_term]definition
```

where:

key_term	is a word, phrase, or Concept for which you want to provide a glossary defirnition
[display_term]	is an optional formal glossary term to display as part of the glossary response. Brackets [ ] are required.
definition	is the glossary definition

Note: The Dictionary Manager displays display terms in the right column of the Alias List display.

Chapter 4

Working with Business Rules

InQuira 6 Business Rules is a separate module that provides access to contextual information about the current user and session for use in determining responses to requests. You can configure the Business Rules module to access information from the current InQuira 6 session and from external repositories.

InQuira 6 collects context information at runtime from configured *Providers* and *Preferences*. Providers are the sources of Preference data. Providers can include the InQuira 6 application itself, web and application servers, and external repositories, such as customer relationship management (CRM) applications.

Preferences are the types of data that you want to access, such as user names, location of residence, and current account information.

InQuira 6 includes a Session Provider that is installed and configured with the application. The Session provider is designed to access only preference data for a given session; it does not store persistent preference data for use in multiple sessions. See the *InQuira 6 Application Guide* for more information on the session provider.

You use Business Rules by:

- configuring Providers and Preferences to access user and session context data, as described in the *InQuira 6 Application Guide*
- referencing context data from within special Rule components called Business Conditions, as described in ***Specifying Business Conditions*** on page 56

Important: You must install and configure the Business Rules module, as described in *InQuira 6 Installation Guide* to use Business Rules functions.

Business Rules Processing

When a user enters a question or request, the Preference Service provides context information for the request according to the configured Providers and Preferences. During request analysis, InQuira 6 uses the context data to evaluate Business Conditions set within Rules.

The Rules Engine evaluates Business Conditions within Rules prior to evaluating language conditions. Only Rules for which the specified Business Conditions are true are then evaluated for matching language conditions.

Note: See ***InQuira 6 Language Analysis*** on page 17 for more information on the request analysis process.

Configuring Preferences and Providers is described in the *InQuira 6 Application Guide*.

Specifying Business Conditions

You specify Business Conditions within Rules using the Business Conditions tab in the Dictionary Manager Rule window. See ***Business Conditions*** on page 33 for more information on how Business Conditions function within Rules.

You can specify multiple conditions for a single Rule.

Important: If you specify multiple Business Conditions, the Rules Engine will evalutate the Rule as true only if all of the Business Conditions are true.

You define Business Conditions by creating one or more expressions consisting of conditions, operators, and values as described in ***Business Condition Components*** on page 56.

You can also set the values of configured context variables dynamically based on information within the user request. The Rules Engine can associate a defined context variable with any aspect of a question that can be represented in IML, as described in Setting Question Context Variables.

Business Condition Components

Business Conditions contain one or more condition expressions. Each condition expression specifies:

- a context variable, as described in ***Business Condition Context Variables*** on page 57
- an operator, as described in ***Business Condition Operators*** on page 58
- a value that the Rules Engine will compare to the current value of the context variable, as specified by the operator

You specify condition expressions in the form:

```
variable | operator | value
```

You can specify multiple Business Conditions within a Rule, and multiple expressions within each Business Condition. The conditions are related by an implicit logical `AND` connector, and the expressions are related by a logical `OR` connector.

Important: If you specify multiple Business Conditions, the Rules Engine will evalutate the Rule as true only if all of its business conditions are true. If you specify multiple condition expressions within a Business Condition, the Rules Engine will evaluate the Business Condition as true if any of the expressions are true.

Business Condition Context Variables

Context variables are properties that correspond the configured Preferences within your application. Some Preferences are packaged with the application; however, you must configure Providers for all Preferences except the date- and time-related Preferences.

The date- and time-related Preferences obtain their values from the current system date and time.

Other pre-defined Preferences, such as user IP address and user name can obtain data from the packaged Session Provider or from a custom Provider configured for your application.

Pre-defined Preferences on page 57 describes the pre-defined Preferences in InQuira 6.

Pre-defined Preferences

The following Preferences are pre-defined within InQuira 6:

Preference	Description
date:now	This Preference is defined as the current system date and time to the millisecond, and can be used with no additional configuration.
date:today	This Preference is pre-defined as 12:00 AM on current system day, and can be used with no additional configuration.
request.baseURL	This Preference is pre-defined to obtain the URL of the request origin. The Preference obtains the URL from a Principal object (`javax.security.Principal`) within the request object. Refer to the *InQuira 6 Application Guide* for information on configuring the Preference Service.
request.referer	This Preference is pre-defined to obtain the URL of the page that originated the request. The Preference obtains the URL from a Principal object (`javax.security.Principal`) within the request object. Refer to the *InQuira 6 Application Guide* for information on configuring the Preference Service.
user.address	This Preference is pre-defined to obtain the IP address associated with the user request. The Preference obtains the IP address from a Principal object (`javax.security.Principal`) within the request object. Refer to the *InQuira 6 Application Guide* for information on configuring the Preference Service.

user.host	This Preference is pre-defined to obtain the host name associated with the user request. The Preference obtains the host name from a Principal object (`javax.security.Principal`) within within the request object. Refer to the *InQuira 6 Application Guide* for information on configuring the Preference Service.
user.name	This Preference is pre-defined to obtain the user name associated with the user request. The Preference obtains the user name from a Principal object (`javax.security.Principal`) within the request object. Refer to the *InQuira 6 Application Guide* for information on configuring the Preference Service.

Business Condition Operators

Operators represent Boolean logic. The Rules Engine will apply the specified operation when comparing the value that it obtains from the Preference Service against the value specified in the condition expression.

The Business Condition window displays natural language phrases that represent the operators available for the specified Preference. Available operators are:

Boolean Operator	Description	Dictionary Manager Example
=	equals	is
!=	does not equal	is not
>	greater than	after
>=	greater than or equal to	is or after
<	less than	before
<=	less than or equal to	is or before

For example, if you specify a condition for customer status:

```
date:today > January 1, 2000
```

the operator > specifies that only requests made after January 1, 2000 will match the Rule.

However, in a similar expression,

```
date.today >= January 1, 2000
```

the operator >= specifies that requests made on or after January 1, 2000 will match the Rule.

Business Condition Values

Values specify the point of comparison for the condition expression. The Rules Engine performs the specified comparison operation using the corresponding value provided by the Preference Service.

For example, the expression:

```
date.today | is or after | July 30, 2003
```

specifies that only requests from users that occur on or after the specified date will match the Rule. The Rules Engine evaluates the condition by comparing the value of `date.today` provided by the Preference Service with the condition value, using the specified operator as the basis of the comparison.

The Business Condition window provides calendar and clock functions for specifying date- and time-related condition values.

Chapter 5

Working with InQuira Match Language

InQuira Match Language (IML) is a language for specifying conditions and actions based on matching words, phrases, and concepts in user requests and application content.

IML is a regular expression language comprising a set of symbols that define various functions and operations. You specify these symbols in combination with words of interest to form IML expressions. The set of characters, their functions, and the rules for using them to create IML expressions are described in ***IML Syntax*** on page 62.

IML provides a flexible means of matching the words in a user request. IML can match a literal word, multiple forms of a word, or even concepts that match or relate to a specified word in some way. You can create IML expressions that match very specifically-worded requests, or that match many general requests that express the same purpose, or intent.

IML expressions are components of Rules; the pre-defined Rules in the Dictionary contain IML expressions that specify both conditions and actions.

You can add and modify IML expressions in question patterns that set conditions and in Search Components that specify search criteria within Actions.

During request processing, the Rules Engine processes the IML within Rule conditions and actions as described in ***IML Processing*** on page 61.

IML Processing

The Rules Engine processes IML during request processing. You can specify IML expressions to match words and phrases in both user requests and unstructured application content.

To match words, phrases, and intents within user requests, you specify IML expressions to create question patterns within Rules. To match words and phrases within unstructured application content, you can:

- use the Dictionary's pre-defined Search Components, which are written in IML
- specify IML expressions within custom Search Components

- specify IML expressions to retrieve specific answers

You can also use IML expressions to restrict the results of SQL queries to configured sources of structured information.

Question Patterns

A Question Pattern is an IML expression that is designed to match the intent of a user request. Question Patterns are one of the components of Rules that you can use to set conditions. Question Patterns range from simple, general expressions to complex, specific ones.

You specify Question Patterns using the various elements of IML to match the words and phrases that occur in user requests.

During request processing, the Rules Engine compares the request to the specified Question Pattern, and evaluates it as true or false. If a Rule is true, the Actions that it specifies will be added to the action list.

To use a Question Pattern within a Rule, you enter an IML expression in the Question Patterns tab in the Conditions section of the Dictionary Manager Rule window, as described in ***Specifying Question Patterns*** on page 135.

Direct IML Expressions

You can specify IML expressions to perform direct retrieval of excerpts from unstructured content. The Rules Engine's default ranking process uses IML to match indexed unstructured content to the user requests. In some cases, you may want to specify an explicit IML expression instead of using the default ranking fuction.

To specify a direct IML expression for document retrieval, you enter an IML expression in the Answer Section tab in the Actions section of the Dictionary Manager Rule window.

IML Syntax

The IML syntax defines multiple elements that you can use to create IML expressions within question patterns, search components, and direct index queries. You create IML espressions by combining the various elements according to the syntax rules.

General IML Syntax Rules

There are no special delimiters to indicate the beginning or end of IML expressions. You generally specify IML expressions in designated fields within Dictionary Manager windows.

You can organize IML expressions in Dictionary Manager fields in any way that you choose. The Rules Engine reads IML expressions from left to right, top to bottom. Line breaks and spaces do not have any function.

Note: The `AND` separator is a functionally identical alternative to the comma separator, and is not an operator. It separates arguments, but does not specify any operation.

Arguments for expressions are delimited by parentheses, and separated by commas, or by the string AND if preferred for clarity. You can create nested expressions using parentheses to delimit the enclosed expressions, and there is no limit to the levels of nesting.

IML Elements

IML supports the following types of elements:

- ***Basic Expressions*** on page 63
- ***Operators*** on page 66
- ***Keywords*** on page 75
- ***Macros*** on page 81
- ***Number Expressions*** on page 78
- ***Variables*** on page 82

Some IML elements allow additional arguments that specify or modify their behavior.

Basic Expressions

Basic IML expressions are the means of specifying semantic matching for words and phrases in user requests and in the indexed content. Basic expressions generally resolve to single units of meaning within user requests or application content. Within indexed application content, single units of meaning occupy a designated position, and are referred to as *offsets*.

The various types of base expressions enable you to match words and phrases as literal character strings, canonical forms that include variations in wordform and punctuation, and concepts that include semantic relations as defined in the Ontology.

You can combine basic expressions with other IML elements to describe and match complex semantic structures. The following basic expressions are valid in IML:

- ***Literal Expressions*** on page 64
- ***Canonical Form Expressions*** on page 64
- ***Concept Expressions*** on page 65

Literal Expressions

A literal expression matches only the exact specified character string. Literal expressions are sensitive to case, punctuation, and spacing.

Syntax:

Enclose the word or phrase within double quotes, in the form:

```
"expression"
```

Example:

The expression…	*matches…*	*and does not match…*
"Cat"	Cat	cat cats

Important: Expressions that include punctuation, such as URLs and URIs, cannot currently be represented as single literal expressions. For example, to match `www.my_company.com` requires the expressions: `'company' '.' 'com'`.

Canonical Form Expressions

Canonical form expressions match the specified string and any variations in inflection or form, such as capitalization, tense, or other valid morphological variations as defined by the Language Analyzer.

Syntax:

Specify canonical form expressions using either of the following forms:

```
expression
‘expression’
```

where:

expression	is a non-quoted string beginning with an upper- or lower- case letter, succeeded by any alphanumeric characters, with no punctuation or spaces
‘expression’	is any alphanumeric string enclosed within single quotes, including punctuation and spaces

Example:

The expresion…	*matches…*	*and does not match…*
cat **'cat'**	Cat cats Cats cat's catty	catalog

Concept Expressions

Concept expressions match occurrences of the specified concept and its synonyms. Concept expressions refer to the sense of the specified concept that is currently defined in the Dictionary. See ***Adding and Modifying Concepts*** on page 143 for information on defining concepts in the Dictionary.

Syntax:

Specify the concept name as defined in the Dictionary, within angle brackets. Concept names consist of three identifiers, separated by periods. The Rules Engine assigns default values to the first two identifiers within a concept reference if they are not specified. You can omit one or both of these identifiers to apply the default values to the specified concept. The complete form of the concept reference is:

```
<pos.domain.headword>
```

Other valid forms are:

```
<type.headword>
<pos.headword>
<headword>
```

where:

pos	specifies the concept's *part of speech*. Valid values are `noun`, `adj` (adjective), `verb`, and `adv` (adverb). This identifier is optional. The default part of speech value is `noun`.
domain	specifies the *domain* that the concept is assigned to. This identifier is optional. The default domain is the application domain name, as specified in the Dictionary.
headword	specifies the concept headword, which is an indicator for the collection of specified specified synonyms and other relationships that define the concept

Example:

The expresions…	match…	and do not match…

`<noun.animal.cat>`
`<animal.cat>`
	Cat
	cats
	kitten
		catalog

Default Concept Reference Values

The following table describes the default values that the Rules Engine assigns when processing concept references.

If you specify…	the Rules Engine assigns…
`<pos.type.headword>`	no default values.
`<type.headword>`	the default part of speech value `noun`.
`<pos.headword>`	the default domain value `application name`, as specified on the Application Instance page of the Configuration and Administration Interface.
`<headword>`	the part of speech value `noun` and the default domain value `application name`, as specified on the Application Instance page of the Configuration and Administration Interface.

Operators

You can specify IML operators to define the portion, or *range*, of a request or document that you want an IML expression to match.

For example, you can specify operators to apply expressions to:

- sentences within documents
- entire documents
- sentences within documents having titles that match a specified expression

Types of Operators

The following types of operators are available in IML:

- range operators

- combining operators

Range operators specify the scope, or amount of surrounding text that will be associated with specified expressions. Examples of ranges include phrases, sentences, and proximity within a specified number of words. Some range operators apply to both conditions (requests) and actions (documents); others apply only to actions.

Combining operators specify operations on specified ranges to further define matching criteria. Examples of combinations include intersections, unions, and overlaps.

Range Operators

Range operators specify the scope of a matched expression. You can use range operators to specify the amount of content that the Rules Engine will associate with the matched expression.

For example, a literal expression `"cat"` will match any occurrences of the string cat. A sentence range operator specifies that this expression will apply to, or match, sentences that contain the string `cat`. A document range operator specifies that this expression will match documents that contain the string `cat`.

You specify range operators as prefixes to the expressions that they operate on. You specify the expressions that the range operator applies to as arguments enclosed within parentheses. You must specify at least one argument for an operator.

You separate arguments with a `,` (comma) or the string `AND`. The expressions that you specify as arguments can be any valid IML expressions.

You can use range operators within IML expressions that specify conditions and actions within Rules; however, not all range operators are valid in both types of expressions.

The following IML range operators apply to both conditions (requests) and actions (documents):

- proximity (`NEAR`) as described in ***Proximity Range Operator*** on page 71

The following IML range operators apply only to actions:

- sentence (`SENT`) as described in ***Sentence Range Operator*** on page 69
- title (`TITLE`) as described in ***Title Range Operator*** on page 69
- subtitle (`SUBTITLE`) as described in ***Subtitle Range Operator*** on page 70
- section (`SECTION`) as described in ***Section Range Operator*** on page 70
- document (`DOC`) as described in ***Document Range Operator*** on page 71
- reference (`REFERENCE`) as described in ***Reference Range Operator*** on page 72

Range Operator Applications

You can use range operators within IML expressions that specify conditions and actions within Rules; however, not all range operators are valid in both types of expressions.

The following IML range operators apply to both conditions (requests) and actions (documents):

- phrase (`PHRASE`). See ***Phrase Range Operator*** on page 68
- proximity (`NEAR`). See ***Proximity Range Operator*** on page 71

The following IML range operators apply only to actions:

- sentence (`SENT`). See ***Sentence Range Operator*** on page 69
- title (`TITLE`). See ***Title Range Operator*** on page 69
- subtitle (`SUBTITLE`). See ***Subtitle Range Operator*** on page 70
- section (`SECTION`). See ***Section Range Operator*** on page 70
- document (`DOC`). See ***Document Range Operator*** on page 71
- reference (`REFERENCE`). See ***Reference Range Operator*** on page 72

Phrase Range Operator

The phrase (`PHRASE`) range operator returns phrases that contain all of the expressions specified as arguments. Phrases are determined by the Language Analyzer analysis process.

`PHRASE` is valid within Rule conditions and actions.

Syntax:

You specify the phrase range operator in the form:

```
PHRASE([expression]{separator | combining operator}{expression}…)
```

where:

`expression`	specifies a valid IML expression as an argument that the range operator will apply to.
`separator`	specifies an argument separator, if required. See ***Range Operators*** on page 67.
`combining operator`	specifies a combining operator, if required. See ***Combining Operators*** on page 73.

Example:

Specify	`PHRASE(<cat>,<hat>)`
To match	phrases containing either of the concepts (synonyms of) `cat` or `hat`

Sentence Range Operator

The sentence (`SENT`) range operator returns sentences that match all of the expressions specified as arguments. Sentences in the Content Store are defined by the Language Analyzer.

`SENT` is valid only within actions.

Syntax:

```
SENT([expression]{separator|combining operator}{expression}…)
```

where:

`expression`	specifies a valid IML expression
`separator`	specifies a separator, if required (`,` or `AND`)
`combining operator`	specifies a combining operator, if required. See ***Combining Operators*** on page 73.

Example:

Specify	`SENT(<cat>,<hat>)`
To match	sentences containing either of the concepts (synonyms of) `cat` or `hat`

Title Range Operator

The title (`TITLE`) range operator returns documents having titles that match all of the expressions specified as arguments. Document titles are identified during content processing, and are stored in a separate index within the Content Store.

`TITLE` is valid only within actions.

Syntax:

```
TITLE([expression]{separator|combining operator}{expression}…)
```

where:

`expression`	specifies a valid IML expression
`separator`	specifies a separator, if required (`,` or `AND`)

combining operator	specifies a combining operator, if required. See ***Combining Operators*** on page 73.

Example:

Specify	`TITLE(<cat>,<hat>)`
To match	documents with titles that contain either of the concepts (synonyms of) `cat` or `hat`

Subtitle Range Operator

The subtitle (`SUBTITLE`) range operator returns document sections having applied subtitles that match all of the expressions specified as arguments. Document subtitles are determined by content processing.

`SUBTITLE` is valid only within actions.

Syntax:

```
SUBTITLE([expression]{separator|combining operator}{expression}…)
```

where:

expression	specifies a valid IML expression
separator	specifies a separator, if required (`,` or `AND`)
combining operator	specifies a combining operator, if required. See ***Combining Operators*** on page 73.

Example:

Specify	`SUBTITLE(<cat>,<hat>)`
To match	document sections having subtitles that contain either of the concepts (synonyms of) `cat` or `hat`

Section Range Operator

The section (`SECTION`) range operator returns document sections that match all of the expressions specified as arguments. Document sections are determined by content processing.

`SECTION` is valid only within actions.

Syntax:

```
SECTION([expression]{separator|combining operator}{expression}…)
```

where:

expression	specifies a valid IML expression
separator	specifies a separator, if required (, or AND)
combining operator	specifies a combining operator, if required. See ***Combining Operators*** on page 73.

Example:

Specify	`SECTION(<cat>,<hat>)`
To match	document sections that contain either of the concepts (synonyms of) `cat` or `hat`

Document Range Operator

The document (`DOC`) range operator returns documents that match all of the expressions specified as arguments. Documents are defined by the content acquisition process.

`DOC` is valid only within actions.

Syntax:

```
DOC([expression]{separator|combining operator}{expression}…)
```

where:

expression	specifies a valid IML expression
separator	specifies a separator, if required (, or AND)
combining operator	specifies a combining operator, if required. See ***Combining Operators*** on page 73.

Example:

Specify	`DOC(<cat>,<hat>)`
To match	documents containing either of the concepts (synonyms of) `cat` or `hat`

Proximity Range Operator

The proximity (`NEAR`) range operator returns a range of words that contains all of the expressions specified as arguments. You specify the range as a parameter, `n`.

`NEAR` is valid within conditions and actions.

Syntax:

`NEAR_n([expression]{separator|combining operator}{expression}…)`

where:

n	specifies the number of words (offsets) that defines the range. Offsets indicate unique index positions, approximately equal to single words.
expression	specifies a valid IML expression
separator	specifies a separator, if required (`,` or `AND`)
combining operator	specifies a combining operator, if required. See ***Combining Operators*** on page 73.

Example:

Specify	`NEAR_5(<cat>,<hat>)`
To match	content having either of the concepts (synonyms for) `cat` or `hat` within a 5 word range

Reference Range Operator

The reference (`REFERENCE`) range operator returns documents having hypertext links to them that match the expressions specified as arguments.

`REFERENCE` is valid only within actions.

Syntax:

`REFERENCE([expression]{separator|combining operator}{expression}…)`

where:

expression	specifies a valid IML expression
separator	specifies a separator, if required (`,` or `AND`)
combining operator	specifies a combining operator, if required. See ***Combining Operators*** on page 73.

Example:

Specify	`REFERENCE(<cat>,<hat>)`
To match	documents having hypertext links pointing to them that contain either of the concepts (synonyms of) `cat` or `hat`

Combining Operators

Combining operators specify operations on expressions or their associated ranges.

You can use combining operators to match an area of content that you define as the result of an operation on two or more specified ranges. For example, you can use an intersection combining operator to return only sentences that include both of two specifed concepts.

You specify combining operators inline, between the expressions that they operate on. The expressions that you combine can be any valid IML expressions.

Combining operators are either binary or have Boolean (`and/or`) semantics.

The following combining operators are valid in IML:

- `IS`, which specifies the intersection of two ranges as described in ***Intersection Operator*** on page 73
- `OR`, which specifies the union of two ranges as described in ***Union Operator*** on page 74
- `ISNT`, which specifies the difference of two ranges as described in ***Difference Operator*** on page 74
- `WITHOUT`, which specifies the difference of two specified ranges as described in ***Offset Difference Operator*** on page 74
- `OVERLAP`, which specifies a range representing the overlap of two ranges as described in ***Offset Intersection Operator*** on page 75

Intersection Operator

The intersection (`IS`) combining operator specifies the intersection of the ranges of the specified expressions.

Syntax:

You specify the intersection combining operator in the form:

```
{expression} IS {expression}
```

where:

`expression`	specifies a valid IML expression

Example:

Specify	`SENT(<cat> IS <hat>)`

To match	sentences that contain the concepts (synonyms of) both `cat` and `hat`

Union Operator

The union (`OR`) combining operator specifies the union of the ranges of the specified expressions.

Syntax:

```
{expression} OR {expression}
```

where:

`expression`	specifies a valid IML expression

Example:

Specify	`SENT(<cat> OR <hat>)`
To match	sentences that contain the either of the concepts (synonyms of) `cat` or `hat`

Difference Operator

The difference (`ISNT`) combining operator specifies the difference of ranges of the specified expressions.

Syntax:

```
{expression} ISNT {expression}
```

where:

`expression`	specifies a valid IML expression

Example:

Specify	`SENT(<cat>) ISNT SENT(<hat>)`
To match	sentences that contain the concept (synonyms of) `cat`, excluding sentences that also include the concept `hat`

Offset Difference Operator

The difference offset (`WITHOUT`) combining operator specifies the range of difference of the specified expressions.

Syntax:

```
{expression} WITHOUT {expression}
```

where:

expression	specifies a valid IML expression

Example:

Specify	`DOC(<cat>) WITHOUT SENT(<hat>)`
To return	documents that contain the concept (synonyms of) `cat`, excluding sentences that include the concept `hat`

Offset Intersection Operator

The overlap (`OVERLAP`) combining operator specifies the ranges representing the offset overlap of all arguments. The overlap operator returns the portion of the text that the specified expressions have in common.

Syntax:

```
{expression}OVERLAP{expression}
```

where:

expression	specifies a valid IML expression that the operator will apply to.

Example:

Specify	`SUBTITLE(cat) OVERLAP hat`
To match	occurrences of `hat` located within sections that have `cat` in their subtitles

Keywords

Keywords perform specific matching functions. You can use keywords to limit matching for an expression to the specified characteristic. For example, you can specify keywords to represent any single word, or to match an expression only if the matching offset is the first word in a document.

You specify keywords inline within the expression that they apply to.

The following keywords are valid in IML:

- `WORD`, which matches any, but exactly one, token as described in ***WORD Keyword*** on page 76.
- `BEGIN`, which matches the first word in a document as described in ***BEGIN Keyword*** on page 76.
- `END`, which matches the last word in a document as described in ***END Keyword*** (on page 77)
- `THIS`, which assigns a concept sense to one or more tokens in IML expressions within the Dictionary as described in ***THIS Keyword*** on page 77

WORD Keyword

The WORD keyword matches any, and exactly one word.

Syntax:

```
[expression] WORD [expression]
```

where:

`expression`	specifies a valid IML expression

Example:

Specify	`this WORD house`
To match	any single word within a matched expression, for example: `this old house` `this red house`
And not match	`this big old house` `this house`

Note: You can modify the WORD keyword to match multiple words using number expressions, as described in ***Number Expressions*** on page 78.

BEGIN Keyword

The BEGIN keyword specifies the beginning of a document, prior to the first word. Use the BEGIN keyword to limit matching for an expression to the first word in a document.

Syntax:

```
BEGIN expression
```

where:

`BEGIN`	specifies to match only the first word in a document
`expression`	specifies any valid IML expression

Example:

Specify	`DOC BEGIN <cat>`
To match	documents having the concept (synonyms of) `cat` as the first word in the body of the document

END Keyword

The END keyword specifies the end of a document, after to the last word. Use the END keyword to limit matching for an expression to the last word in a document.

Syntax:

```
END expression
```

where:

`END`	specifies to match only the first word in a document
`expression`	specifies any valid IML expression

Example:

Specify	`DOC END <cat>`
To match	documents having the concept (synonyms of) `cat` as the last word in the body of the document

THIS Keyword

The THIS keyword is valid only in IML expressions within concept definitions in the Dictionary. The THIS keyword specifies a part of an IML expression that represents the defined concept. You can use THIS to limit the part of expression that the Rules Engine will use as the concept in subsequent operations.

Syntax:

You specify the THIS keyword in the form:

```
expression=THIS
```

where:

`THIS`	specifies the potion of the expression to use as the concept in subsequent operations
`expression`	specifies any valid IML expression that defines the concept

Example:

Specify	`(online OR virtual) NEAR_5 banking=THIS`
To define	a concept `<online banking>`

That matches	portions of text like:
	`banking online is fun and easy...`
	`try banking the virtual way with our...`

And specifies that	the Rules Engine will use the word `banking` in subsequent operations on this concept

Number Expressions

Number expressions specify a number or range of occurrences to match for the expression they apply to. You can use number expressions to apply numeric ranges to any valid IML expressions.

For example, you can specify to match one or more occurences of an expression, or up to five occurrences of a specified expression.

The following number expressions are valid in IML:

- simple range number expressions as described in ***Simple Range*** on page 79
- ascending range number expressions as described in ***Ascending Range*** on page 80
- descending range number expressions as described in ***Descending Range*** on page 80

General Rules for Specifying Number Expressions

You specify number expressions as a single range of integers separated by a hyphen and enclosed within parentheses. Number expressions follow the expression they apply to.

Syntax:

```
expression(number_expression)
```

where:

`expression`	specifies a valid IML expression
`number_ expression`	specifies a valid number expression of the following type: • ***simple range*** on page 79 • ***ascending range*** on page 80 • ***descending range*** on page 80

Simple Range

The simple range number expression specifies a range that spans the specified lower and upper boundaries.

Syntax:

```
expression(n-m)
```

where:

`n`	specifies the lower limit of the range
`m`	specifies the upper limit of the range. The value of m must be greater than the value of n
`expression`	specifies any valid IML expression

Example:

Specify	`<hat>(2-3)`

To match	occurrences of the concept (synonyms of) `hat` in a series of two or three:
	`hat hat`
	`hat hat hat`

And not match	single occurrences of the term `hat`

Note: In the match example, the Rules Engine would produce a total of four matches:

- a match for the series of two `hats` on the first line: `{hat hat}`
- a match for the first series of two `hats` on the second line: `{hat hat} hat`
- a match for the second series of two `hats` on the second line: `hat {hat hat}`
- a match for the series of three `hats` on the first line: `{hat hat hat}`

Ascending Range

The ascending range number expression specifies all integers greater than or equal to the specified lower boundary.

Syntax:

```
expression (n-) expression
```

where:

n	specifies the lower limit of the range
expression	specifies any valid IML expression

Example:

Specify	`cat WORD(2-) hat`

To match	occurrences of the specifed expressions `cat` and `hat` having intervening words within the specified range or two or more:
	`cat and the hat`
	`cat and the red hat`

And not match	occurrences of the specifed expressions `cat` and `hat` having intervening words beyond the specified range or two or more: `cat and hat`

Descending Range

The descending range number expression specifies all integers less or equal to than the specifed upper boundary, including 0.

Syntax:

```
expression(-n)
```

where:

n	specifies the lower limit of the range
expression	specifies any valid IML expression

Example:

Specify	`cat WORD(-2) hat`

To match	occurrences of the specifed expressions `cat` and `hat` having intervening words within the specified range or two or fewer:
	`cat hat`
	`cat and hat`
	`cat and the hat`

And not match	occurrences of the specifed expressions `cat` and `hat` having intervening words beyond the specified range or two or fewer: `cat and the red hat`

Macros

Macros are assigned character substitutions for commonly specified IML expressions. You can use macros to specify 0 or more occurrences of any word, and 1 or more occurrences of any word. The following macros are defined in IML:

*

Specifies 0 or more occurrences of a word. See ***Zero or More Words Macro*** on page 81

+

Specifies 1 or more occurrences of an expression. See ***One or More Words Macro*** on page 82

You specify macros inline within IML expressions, in the form:

```
{expression}macro{expression}
```

Zero or More Words Macro

You can specify the macro * to match occurrences of zero or more words, in the following form:

```
{expression}*{expression}
```

where:

+	matches occurrences of one or more words
`expression`	specifies any valid IML expression

Note: The * macro can be expressed in IML as the following keyword and number expression: `WORD(0-)`

One or More Words Macro

You can specify the macro + to match occurrences of one or more words, in the following form:

```
{expression}+{expression}
```

where:

+	matches occurrences of one or more words
expression	specifies any valid IML expression

Note: The + macro can be expressed in IML as the following keyword and number expression: `WORD(1-)`

Variables

A variable is a symbol that represents a contiguous set of words in a user request. Variables are a method of associating a part of the user request with an expanded (more general) or reduced (more specific) set of meanings.

There are local and global variables. Local variables apply only within the Rule in which they are specified. Global variables apply within the Rule in which they are set, and also within subsequent Rules.

The Rules Engine processes variables during request processing. When the Rules Engine evaluates a Rule as true, it sets any variables specified within the Rule. Once a variable is set, its value is substituted when it is referenced from another expression. Local variables can be referenced only by expressions within the same Rule (but not from within the expression in which they are set). Global variables can be referenced by any subsequent Rules.

Note: You cannot refer to a global variable in a Rule that preceeds the Rule in which it is set. The Dictionary Manager automatically checks global variables for validity when you update Dictionary data, and will issue a warning if this occurs.

You use variables by:

- specifying them within IML expressions as described in ***Specifying Variables*** on page 82
- assigning their values as described in ***Assigning Values to Variables*** on page 83
- referring to them as described in ***Referring to Variables*** on page 83

Specifying Variables

You specify a variable within an IML expression in the format:

```
(expression)=variable
```

or

```
(expression)=#variable
```

where:

#	specifies a global variable, which applies to all subsequent rules. Omitting the # prefix specifies a local variable, which applies only to the current rule.
variable	can be any alphanumeric string. The first character in the string must be a letter (alpha) character. Letter characters can be upper and lower case. If you specify only upper case characters, you can also use the – (hyphen) character. Global variables can also include the characters: • _ (underscore) • – (hyphen, available only when using all uppercase characters) • $ • % You can specify the same string for multiple variables. If you specify the same string for multiple global variables, the Rules Engine adds each assignment as it is processed.

Example:

```
expression=A
```

Assigning Values to Variables

You assign the value of a variable using Variable Instantiation Language (VIL) functions. When you set a variable within an IML expression, you use various VIL functions to determine its value, or *instantiate* it in the context of an individual user request. You specify these functions using VIL expressions, usually within a Rule action. VIL expressions contain one or more function calls, and each function call has one or more optional parameters, as described in ***VIL Syntax*** on page 84.

Referring to Variables

You can include VIL expressions to refer to variables from within index queries, custom content, SQL queries, or other expressions. You specify references to variables within curly brackets { }, in the form:

```
{VIL_expression}
```

where { } delimits VIL expressions within surrounding text such as IML expressions or custom content.

Example:

You can use VIL expressions to refer to variables from within custom content specified in the answer section of a Rule:

```
You can buy a {VIL_expression_1} today for {VIL_expression_2}.
```

where:

`VIL_expression_1`	refers to a variable set in a Rule condition that resolves to a product name or type of product mentioned in a user request that matched a `price of product` Rule.
`VIL_expression_2`	refers to a variable instantiated from a database query for the price of the product mentioned in the user request.

The instantiated statement displayed by the User Interface might be:

You can buy the *Model 500 washer/dryer* today for $*900*.

You can also specify a variable to resolve to a valid IML expression, which the Rules Engine then evaluates as an action.

The Variable Instantiation Language (VIL)

InQuira 6 uses a special language, called the Variable Instantiation Language (VIL) to resolve, or *instantiate*, variables set within Rules. The Rules Engine uses VIL to generate and manipulate strings that ultimately determine values for variables.

You use VIL by specifying expressions within Rule conditions and actions. Within conditions, you can use VIL expressions to describe IML conditions based on request content. Within actions, you can specify VIL expressions within search components, custom content, or other text fields.

You specify VIL expressions in the form described in ***VIL Syntax*** on page 84. The Rules Engine processes VIL expressions as described in ***VIL Function Processing*** on page 85.

VIL Syntax

The Variable Instantiation Language consists of functions, parameters, operators, and delimiters that you combine to create valid VIL expressions.

You specify VIL expressions in the form:

```
{variable}{operator}function([parameter=value, parameter=value,
...])
```

where:

`variable`	is an optional variable passed to the function for instantiation
`operator`	is a valid VIL operator, as described in ***VIL Operators and Delimiters*** on page 87
`function`	is a valid VIL function, as described in ***VIL Functions*** on page 88
`parameter`	is a valid parameter of the function. Parameters are described within the function descriptions to which they apply.
`value`	is a specifed value of the parameter

VIL Function Processing

VIL functions take information as input, and pass information as output. They can also refer to various services and data. Input is in the form of sets of strings, which are associated with words in user requests, as described in ***VIL Function Input*** on page 85.

Most VIL functions manipulate input according to specified parameters before passing them as output.

The Rules Engine generally applies functions to the input strings in sequence; however, it sometimes applies functions only to some strings, or to a combination of strings, depending on the functions and parameters that are specified.

VIL functions can refer to any of the strings in any of the input sets (corresponding to any word in the question) as they calculate their output.

VIL Function Input

The input to VIL functions is a set. Each input set refers to a wordform in the user request. The input set is composed of subsets, and the subsets are sets of strings.

The literal string contains disambiguation information for the wordform occurring in the user request. For each disambiguated wordform, the set of strings will be a single string, which is a normalized form (token) of the word, as determined by the Language Analyzer.

For example, consider a user request:

```
What does a hot dog cost?
```

and a Rule that assigns a variable `A` to the portion of the request `hot dog cost`.

The string-set associated with the words `hot dog` would contain a single string, `hot dog`. The string-set associated with the word `cost` would contain a single string, `cost`.

If, for the purpose of this example, the Rules Engine sends these strings to the following `MORPH` function, which generates morphological variants, as described in ***The Morphology Function*** on page 92:

```
A-> MORPH(FORMS=OP)->STRING()
```

the MORPH function will generate the strings in their original forms (O) and in their plural form (P), resulting in the string sets:

```
hot dog
hot dogs
```

and

```
cost
costs
```

Note: The strings in these sets are always wordforms, and always occur in the Ontology, even if as the `unknown` wordform.

VIL Function Output

VIL function output is a set of sets of strings, where each of the higher order of sets corresponds to a wordform in the user request, except for the output of the STRING function.

The output of the STRING function is a single string that concatenates all of its input strings. The STRING function is always the last function in a VIL expression, either as an implied function, using the default values, or using explicitly specified values. See ***The String Function*** on page 97 for more information.

Combining VIL Functions

You can specify multiple functions within a VIL expression. The Rules Engine performs the specified functions serially, in the order that it reads them, using the output from the preceding function as the input for the next.

Input and output strings may not always apply to the specified functions in a VIL expression.

Note: A non-applicable string is different than an applicable string that produces a null output. See ***Non-Applicable and Null Output Strings*** on page 87 for an explanation of the implications of this difference.

If a function receives a non-applicable input string, it will either pass that string, unprocessed, as input to the next function, or not, depending on the specified operator. Operators specify the ways in which function outputs will be used as input by subsequent functions, as described in ***VIL Operators and Delimiters*** on page 87.

You can specify any number of VIL functions and operators in any order, except the STRING function. The STRING function must be the last specified function in a VIL expression. The Rules Engine assumes the STRING function's default values unless you explicitly specify the STRING function within a VIL expression.

VIL Operators and Delimiters

VIL uses the following operators to specify how the Rules Engine will pass data between specified functions.

`->`	specifies to pass function output as input to a subsequent function. For example: `function_1(parameter) -> function_2(parameter)` specifies that the Rules Engine will send the output of `function_1` as input to `function_2`.
`||`	specifies to pass null function output as input to a subsequent function. For example: `function_1||function_2` specifies that the Rules Engine will send only the null output of `function_1` as input to `function_2`.

Non-Applicable and Null Output Strings

In VIL processing, there is an important difference between strings that do not apply to functions and applicable strings that return null values. The Rules Engine passes non-applicable strings to the next specified function, depending on the type intervening operator. The operators within VIL functions process strings that return null values differently than non-applicable strings, as described in the following table:

If a string…	*And the operator is…*	*The Rules Engine…*
does not apply	`->` or `||`	passes the string directly to the next function
returns a null value	`->`	passes the string resulting from the function (null) to the next function
returns a null value	`||`	no string to the next function and uses the latter function on the input that returned a null value for the earlier function

For example, the function:

```
ALIAS(WLIST="skipwords")
```

specifies that every string will be compared to a list of skipwords. The skipwords list refers to a data table that specifies a replacement string value of no-output.

If the next function is specified with the `||` combining operator, as follows:

```
ALIAS(WLIST="skipwords") || function_2(parameter)
```

then:

- strings that have no entry in the skipwords data table will be sent to `function_2`
- strings that appeared in the skipwords table, and therefore produced no output will not be sent to `function_2`

VIL Parameters

You can specify parameters to control the behavior of VIL functions. Each parameter has a set or range of valid values.

For example, the STRING function has the following parameters that you can use to add additional contents to the string that is passed as input:

- BEGIN
- END
- SEP

Each of the parameters accepts a string as its value, so that if you specify:

```
cat hat -> STRING(BEGIN=">>> ", END=" <<<", SEP=" --- ")
```

The Rules Engine will produce the string:

```
>>> cat --- hat <<<
```

VIL Functions

You specify VIL functions within VIL expressions, along with the various parameters and operators that control how the Rules Engine processes the functions and passes data between them. The available VIL functions are:

- ***ALIAS*** see "The Alias Function" on page 89
- ***INHERIT*** see "The Inherit Function" on page 90
- ***MORPH*** see "The Morphology Function" on page 92

- ***NOVAR*** see "The Novar Function" on page 93
- ***ONTOLOGY*** see "The Ontology Function" on page 93
- ***REPLACE*** see "The Replace Function" on page 96
- ***STRING*** see "The String Function" on page 97
- ***TEXT*** see "The Text Function" on page 98
- ***USERDATA*** see "The User Data Function" on page 103

The Alias Function

The ALIAS function replaces strings based on configured Alias Lists that are stored in the Dictionary. Alias Lists enable transformations, such as conversion of text strings to corresponding database values to support SQL queries. For more information on Alias Lists, see ***Working with Alias Lists*** on page 52.

Syntax:

```
ALIAS(LIST="list_name",[CASESENSITIVE=value])
```

where:

LIST	specifies the name of the list, as defined in the Dictionary, enclosed in double quotes. There is no default value.
CASESENSITIVE	specifies that the Rules Engine will only perform transformation only when capitalization of terms matches. This parameter is optional. The default is `FALSE`.

Example:

The following expression:

```
ALIAS(LIST="skipwords")
```

specifies that every input string will be compared to the entries in the skipwords alias list for processing.

Example:

Consider a Rule that assigns the variable `A` to the word `Accounting`:

```
A="Accounting"
```

and an Alias List named `PROGRAMS` that specifies the alias entry:

```
"Accounting", "ACCT"
```

If a VIL function call within a Rule action specifies:

```
A->ALIAS(LIST="PROGRAMS")
```

The Rules Engine uses the VIL functions to transform the value of the variable `A` to the value `ACCT`.

The Inherit Function

The INHERIT function specifies inheritance criteria for Search Components. See ***Working with Search Components*** on page 44 for more information on Search Components and how they are assigned. The INHERIT function creates a string containing all assigned Search Components.

Syntax:

```
INHERIT(TYPE=value,SUBTYPE=value,[SCORE=number],[SEP=string],[EXACT=
value],[NODUPS=value])
```

where:

TYPE	specifies the required type of search components to be inherited. Possible values are: **Index** Index Search Components use IML expressions to define criteria for searching the indexes of unsructured content in the Content Store. **SQL** SQL Search Components use SQL queries to define criteria for retrieving information from configured structured data sources. The default value is `ALL`. See ***Search Component Types*** on page 45 for more information.
SUBTYPE	specifies a Search Component subtype of search components to be inherited. The default value is `ALL`. See ***Search Component Subtypes*** on page 46 for more information.
SCORE	specifies the allowed score range of search component levels to be inherited. The default value is `0`.
SEP	specifies an optional separator string to insert between output strings. This parameter is optional. The default value is null (`""`).
EXACT	specifies whether Search Components are inherited only if the variable range is also the exact range over which the Search Component is assigned. See ***Search Component Ranges*** on page 45 for information on search components ranges. Valid values are true and false. True specifies that the variable range and the search component range must match exactly. The default value is `FALSE`.

NODUPS	specifies whether identical search component levels will be inherited more than once. The default value is `FALSE`.

Example:

```
A->INHERIT(SCORE=”20”)
```

creates a string consisting of all search components already assigned to A, and having a score of 20.

Note: The Evaluator translates importance and quality values into absolute scores associated with each entry in a Search Component. See ***Scoring and Ranking Responses*** on page 21 for more information.

When the Search Component covers only one string set, the inherited Search Component uses exactly that string set.

In cases where the Search Component covers more than one string set, the string sets are combined, separated by the value of the SEP argument.

Example:

Consider a query:

```
april shower
```

that has a variable A assigned to it, and that a search component is assigned to the portion `"april"`:

```
(apr OR april)
```

with a score of `10`.

Consider the VIL expression:

```
{{april} {shower}} -> INHERIT(SCORE=10) || MORPH(FORMS=”O”)
```

which produces the string:

```
(apr OR april) shower
```

If the original string has an additional search component assigned to the portion `april shower`:

```
(rain)
```

that has a score of `10`:

```
{{april} {shower}} -> INHERIT(SCORE=10) || MORPH(FORMS=”O”) =>
{{(rain)} {}}
```

Example:

Consider a Rule that assigns the search component variable `A` to the word `Accounting`:

```
A="Accounting"
```

and the corresponding search components:

```
1000 ("Accounting")  IS (<verb.possession:account>
1000 <noun.possession:account2>
1000 <noun.state:account>
```

If a VIL function call within a Rule action specifies:

```
A-> INHERIT( EXACT=TRUE,SCORE=1000,SEP = ”) OR (“,TYPE =
“index”,SUBTYPE = “standard” )
```

The Rules Engine uses the VIL functions to transform the value of the variable A to:

```
(("Accounting") IS (<verb.possession:account> OR
<noun.possession:account2> OR <noun.state:account>))
```

The Morphology Function

The MORPH function creates specific morphological inflections of wordforms. The MORPH function can create the following forms:

S	generates simple past and past participles
P	generates plural forms
O	generates original wordforms
I	generates infinitive (stem) wordforms
G	generates gerund (-ing) wordforms
A	generates actor (-er) wordforms

Note: If the MORPH function is not called, the Rules Engine uses the stem of each wordform, as determined by the Language Analyzer.

Syntax:

```
MORH([FORMS=“[S|P|O|I|G|A]*”],[SEP=<string>])
```

where:

FORMS	specifies one or more of the valid morphological forms. The default is `O`, which generates the original wordform used in the request.
SEP	specifies an optional separator string to insert between output strings. This parameter is optional. The default value is `OR`.

Example:

Consider a Rule that assigns the search component variable `A` to the word `Accounting`:

```
A="Accounting"
```

and a VIL function call within a Rule action to generate the infinitive and plural forms of the stem word:

```
A->MORPH(FORMS="IP")
```

The Rules Engine processes the MORPH function and generates:

```
account OR accounts
```

The Novar Function

NOVAR is a special function designed to handle cases in which a variable's value is null. You specify the NOVAR function in the form:

```
NOVAR([ACTION="DROP" | <string>])
```

where:

ACTION	specifies the action to perform. Valid values are a specified string to use as output, and `DROP`, which specifies to stop processing and discard this expression. The default is to output a null string (`""`).

Example:

```
username -> NOVAR(ACTION="Valued Customer")
```

generates the string `Valued Customer` when there is no value associated with the variable `username`.

Example:

Consider a Rule that assigns a variable `A` with a null (`""`) value.

```
A=""
```

and a VIL function call within a Rule action to assign the string `Default Value` to the null variable:

```
A->NOVAR(ACTION="<Default Value>")
```

The Rules Engine uses the NOVAR function to generate:

```
Default Value
```

The Ontology Function

The ONTOLOGY function generates Ontology entities by traversing the Ontology hierarchy using a starting point defined by the input, and a direction and distance defined by the ONTOLOGY function parameters.

Syntax:

```
ONTOLOGY([EXACT=value],[CONTAINED=value],[TRAVERSAL=value],[SEP=<str
ing>])
```

where:

EXACT	specifies to use only concepts that encompass the exact range of tokens as the instantiated variable; the concept is therefore an exact match for the variable. Valid values are `TRUE` and `FALSE`. The default is `FALSE`.
CONTAINED	specifies to use concepts that are contained within the range of tokens that make up the instantiated variable, for example a variable consisting of four tokens where tokens 2 and 3 denote a concept.
TRAVERSAL	specifies the option that determines the direction and distance that the function will travel to generate output. See ***Ontology Traversal Functions*** on page 94 for valid traversal options.
SEP	specifies an optional separator string to insert between output strings. This parameter is optional. The default value is null (`""`).

Example:

Consider a Rule that assigns a variable `A` with a value `Account`.

```
A="Account"
```

and a VIL function call within a Rule action to generate the immediate parent noun concepts:

```
A->ONTOLOGY(EXACT=TRUE,CONTAINED=TRUE,TRAVERSAL="isaUpOneNoun",SEP="
OR " )
```

The Rules Engine uses the ONTOLOGY function to locate the specified Ontology relations and generate the parent concepts for `Account`:

```
(<noun.communication:statement1> OR <noun.state:account>)
```

Ontology Traversal Functions

The Ontology traversal functions travel a specified number of links of a specified type within the Ontology hierarchy.

The Ontology traversal functions take a set of string-sets (wordforms) as input, locate the relevant link type for the specified function, and return the set of wordforms associated with the specified traversal options.

Option	*Description*
isaDnAllNoun	This option takes concepts as input and returns all child noun concepts that descend from the input concept.
isaSynDnAllNoun	This option takes concepts as input and returns the noun synonyms of all of their children.

isaDnAllVerb	This option takes concepts as input and returns all child verb concepts that descend from the input concept.
isaSynDnAllVerb	This option takes concepts as input and returns the verb synonyms of all of their children as a set of stem words.
isaUpOneNoun	This option takes concepts as input and returns the immediate parent noun concepts.
isaSynUpOneNoun	This option takes concepts as input and returns the immediate parent noun concepts as sets of stem words.
isaDnOneNoun	This option takes concepts as input and returns the immediate child noun concepts.
isaDnTwoNoun	This option takes concepts as input and returns the two levels of child noun concepts.
isaSynDnOneNoun	This option takes concepts as input and returns the immediate child noun concepts as sets of stem words.
isaSynDerDnAllNoun	This option takes concepts as input and returns the noun synonyms of all of their children as a set of stem words.
attrUpOneAdj	This option takes concepts as input and returns the immediate parent adjective concepts, for example, `hot -> temperature`.
attrSynUpOneAdj	This option takes concepts as input and returns the immediate parent adjective concepts as a set of stem words.
attrUpOneNoun	This option takes concepts as input and returns the immediate parent noun concepts.
partofUpOneNoun	This option takes concepts as input and returns the immediate *part-of* parent noun concepts.
partofSynUpOneNoun	This option takes concepts as input and returns the immediate *part-of* parent noun concepts as sets of stem words.
partofSynDnAllNoun	This option takes concepts as input and returns the noun synonyms of all descendant concepts having *part-of* relations as a set of stem words.
partofDnAllNoun	This option takes concepts as input and returns all of the descendant concepts having *part-of* relations.
synDer	This option takes concepts as input and returns their synonyms as a set of stem words.

concClstr	This option takes concepts as input and returns all concepts that share a cluster relation. If an input concept belongs to multiple clusters, this option returns all members of all clusters.
thisConcept	This option returns the concept from which the traversal started.

The Replace Function

The REPLACE function replaces the original string if the string or a substring match the specified pattern.

Syntax:

You specify the REPLACE function in the form:

```
REPLACE(FIND=<string>,REPLACE=<string>,CASESENSITIVE=value,CONTAINS=
value)
```

where:

FIND	specifies the string or substring to match. This is a required parameter. The default is `""`.
REPLACE	specifies a replacement string for the matched string or substring. This is a required parameter. The default is `""`.
CASESENSITIVE	specifies whether the Rules Engine will match strings only if upper- and lowercase characters match. Valid values are `TRUE` and `FALSE`. The default is `FALSE`.
CONTAINS	specifies whether to match a substring of the input string. Valid values are `TRUE` and `FALSE`. The default is `FALSE`.

Example:

Consider a Rule that assigns the variable `A` to the string `Account`.

```
(Account)=A
```

and a VIL function call within a Rule action to replace strings containing the substring `Acc` with the string `ACCT`:

```
A->REPLACE( FIND="Acc", REPLACE="ACCT", CONTAINS=TRUE )
```

The Rules Engine uses the REPLACE function to produce:

```
ACCT
```

The String Function

The STRING function concatenates a set of input strings to create a single string. You can also specify parameters to:

- add strings to the beginnning of the input strings
- add strings to the end of the input strings
- separate sets of input strings
- remove redundant strings from the final concatenated string

The STRING function is implied in any VIL expression. You do not need to specify the STRING function unless you want to use parameters other than the defaults.

You specify the STRING function in the following form:

```
STRING(BEGIN='string' | SEP='string' | END='string' | NODUPS=value |
LOWERCASE=value | ALWAYSAFFIX=value, |ESCAPE=<string>)
```

where:

BEGIN	specifies to insert a specified string at the beginning of each substring in the concatenated string. The default value is `"("`.
SEP	specifies an optional separator string to insert between output strings. This parameter is optional. The default value is `""`.
END	specifies to insert a specified string at the end of each substring in the concatenated string. The default value is `")"`.
NODUPS	specifies whether to remove redundant strings within the concatenated string. Valid values are `TRUE`, which specifies to remove redundant strings, and `FALSE`, which specifies to preserve redundant strings. The default value is `TRUE`.
LOWERCASE	specifies to transform the output string to lowercase characters. Valid values are `TRUE` and `FALSE`. The default is `FALSE`.
ALWAYSAFFIX	specifies whether the Rules Engine will add the specified BEGIN and END values to null strings. Valid values are `TRUE` and `FALSE`. The default is `FALSE`.
ESCAPE	specifies a character for escape processing, so that characters immediately following the ESCAPE character will be processed as literal strings, not as syntax elements.

Example:

Consider a Rule that assigns the variable A to the string `cat dog`.

```
(cat dog)=A
```

and a VIL function call within a Rule action to add the specified elements to the input string:

```
A->STRING(BEGIN=">>>", END="<<<", SEP="---")
```

The Rules Engine uses the STRING function to produce::

```
>>> cat --- dog <<<
```

The Text Function

The TEXT function performs text editing operations on the input string according to specified parameters.

Syntax:

You specify the TEXT function in the form:

```
TEXT([ARG1=<string>],[ARG2=<string>],{OPERATION=operator})
```

where:

ARG1	specifies an argument to the specified operation.
ARG2	specifies an argument to the specifed operation.
OPERATION	specifies one of the following operations:

- ***capitalize*** see "The Capitalize Operation" on page 99
- ***concat*** see "The Concatenate Operation" on page 99
- ***count*** see "The Count Operation" on page 100
- ***lowercase*** see "The Lowercase Operation" on page 100
- ***replace*** see "The Replace Character Operation" on page 100
- ***sort*** see "The Sort Operation" on page 101
- ***tokenize*** see "The Tokenize Operation" on page 101
- ***trim*** see "The Trim Operation" on page 102
- ***unique*** see "The Unique Operation" on page 102
- ***uppercase*** see "The Uppercase Operation" on page 103

Example:

Consider a Rule that sets the variable `A` to the string:

```
A="This is an Account"
```

and a VIL function call within a Rule action to replace the matching substring `i` with the specified string `I`:

```
A->TEXT(ARG1="i", ARG2="I", OPERATION="replacechar")
```

The Rules Engine uses the TEXT function to edit the input string and produce:

```
ThIs Is an Account
```

The Capitalize Operation

The Capitalize operation of the TEXT function capitalizes (transforms the initial character to uppercase) each word within the input string. The Capitalize operation has no arguments.

Example:

```
TEXT(OPERATION="capitalize")
```

Input String:

```
This is the Answer
```

Result:

```
This Is The Answer
```

The Concatenate Operation

The Concatenate (`concat`) operation of the TEXT function adds specified strings as prefixes or suffixes to each word in the input string. The Concatenate operation has the following arguments:

Argument	*Description*
ARG1	specifies a prefix string to add to each word in the input string
ARG2	specifies a suffix string to add to each word in the input string

Example:

```
TEXT(ARG1="PREFIX", ARG2="SUFFIX", OPERATION="concat")
```

Input String:

```
This is the Answer
```

Result:

```
PREFIXThisSUFFIX PREFIXisSUFFIX PREFIXtheSUFFIX PREFIXAnswerSUFFIX
```

The Count Operation

The Count operation of the TEXT function transforms the input string to an integer corresponding to the number of tokens (words) in the string. The Count operation has no arguments.

Example:

```
TEXT(OPERATION="count")
```

Input String:

```
This is the Answer
```

Result:

```
4
```

The Lowercase Operation

The Lowercase operation of the TEXT function transforms the input string to lowercase characters. The Lowercase operation has no arguments.

Example:

```
TEXT(OPERATION="lowercase")
```

Input String:

```
This is the Answer
```

Result:

```
this is the answer
```

The Replace Character Operation

The Replace Character (`replacechar`) operation of the TEXT function replaces a character specified as the initial argument with an alternate specified as the second argument. The Replace Character operation has the following arguments:

Argument	*Description*
ARG1	specifies the character to replace within the the input string
ARG2	specifies a character to replace the specified character with

Example:

```
TEXT(ARG1="s", ARG2="t", OPERATION="replacechar")
```

Input String:

```
This is the Answer
```

Result:

```
Thit it the Antwer
```

The Sort Operation

The Sort operation of the TEXT function sorts the tokens (words) within the input string in alphabetical order. The Sort operation has the following arguments:

Argument	Description
ARG1	`true` specifies case-sensitive sorting. `false` specifies case-insensitive sorting. `false` is the default. Case-sensitive sorting is conducted according to ASCII sort standards (uppercase letters precede lowercase letters).
ARG2	`reverse` specifies reverse (z-a) sorting. `normal` (or any value other than reverse) specifies a-z sorting.

Example:

```
TEXT(OPERATION="sort")
```

Input String:

```
This is the Answer
```

Result:

```
Answer is the This
```

Example:

```
text(ARG1="true", OPERATION="sort")
```

Input String:

```
This is the Answer
```

Result:

```
Answer This is the
```

The Tokenize Operation

The Tokenize TEXT operation tokenizes the input string. You can use the Tokenize operation to break VIL values into their constituent parts. For example, the Tokenize operation can separate a single multiword value (InQuira Corporation) into separate words (InQuira, Corporation) to enable further processing of the individual words. The Tokenize operator has the following argument:

Argument	Description
ARG1	specifies a character or string to use as a delimiter between tokens. The default is `~!@#$%^&*()_+=[]\\{}|;':\",./<>?\t\r\n`, which tokenizes input at any of the default characters.

Example:

```
TEXT(ARG1=". ",OPERATION="tokenize")
```

Input String:

```
www.inquira.com
```

Result:

```
www inquira com
```

The Trim Operation

The Trim operation of the TEXT function removes white space from before and after the string.

Example:

```
TEXT(OPERATION="trim")
```

Input String:

```
"   This is the Answer   "
```

Result:

```
"This is the Answer"
```

The Unique Operation

The Unique operation of the TEXT function removes duplicate tokens (words) within the input string. The Unique operation has no arguments.

Example:

```
TEXT(OPERATION="unique")
```

Input String:

```
This is the the Answer
```

Result:

```
This is the Answer
```

The Uppercase Operation

The Uppercase operation of the TEXT function transforms the input string to uppercase characters. The Uppercase operation has no arguments.

Example:

```
TEXT(OPERATION="uppercase")
```

Input String:

```
This is the Answer
```

Result:

```
THIS IS THE ANSWER
```

The User Data Function

The USERDATA function returns the value of a configured preference.

Syntax:

You specify the USERDATA function in the form:

```
USERDATA(KEY=<string>)
```

where:

KEY	specifies the key configured in the application associated with the defined preference. The default key is `Date`.

Example:

Consider a preference that obtains the user's name from a login process:

```
customer_name = Edwin
```

a VIL function call within a Rule action to retrieve that preference value:

```
A->USERDATA(KEY=customer_name)
```

and a managed response that includes the phrase:

```
Thank you for asking, <customer_name>
```

The Rules Engine uses the USERDATA function to transform the variable to the value associated with key `customer_name` in the Preference Service:

```
Thank you for asking, Edwin
```

Chapter 6

Using the Dictionary Manager

The Dictionary Manager is an application within the Language Workbench that you use to customize and maintain the Dictionary for your application. The Dictionary Manager provides access to all of the Dictionary objects, including the Rules, Concepts, and Alias Lists defined for your application.

You use the Dictionary Manager's various menus, tools, and windows to create and modify Dictionary objects.

To access the Dictionary Manager, you must have the Workbench installed, have a valid user ID and password, and have the appropriate permissions, as described in ***Managing User Accounts*** on page 129. To update the Dictionary for your application, your Dictionary Manager must be configured to access the Central Dictionary, as described in *InQuira 6 Application Guide*.

The Dictionary Manager uses a copy of the Dictionary, which you can then modify and test prior to committing changes back to the Central Dictionary, as described in ***Updating Dictionary Data*** on page 149.

Accessing the Dictionary Manager

You can access the Dictionary Manager:

- from the Workbench Launcher
- from other Workbench applications, such as the User Manager

To access the Dictionary Manager from the Workbench Launcher:

- enter a valid User name and password
- select Dictionary Manager from the Application dropdown menu

To access the Dictionary Manager from other Workbench applications:

- double-click on the Dictionary Manager icon, displayed in the lower right corner of the main application window

The Dictionary Manager Main Window displays, and the Find Definitions window displays within the Main Window.

The Dictionary Manager Main Window

When you launch the Dictionary Manager, the Dictionary Manager Main Window displays the menu bar and tool bar, as described in ***Dictionary Manager Menus*** on page 106, and the Find Definitions window, as described in ***Find Definitions Window*** on page 110.

Dictionary Manager Menus

The Dictionary Manager Menu bar contains the following menus:

File

The File menu contains the Rule, Concept, Alias List, Export Domain Group, and Exit options. See ***The File Menu*** on page 106 for more information on using these options.

Edit

The Edit menu contains the Find Definitions, All Rules, Concept Tree, and All Alias List options. See ***The Edit Menu*** on page 107 for more information on using these options.

View

The View menu contains the Rule creation preferences and Cascade all windows options. See ***The View Menu*** on page 108 for more information on using these options.

Tools

The Tools menu contains the Commit work to central repository, Update from central repository, Conflicts, Check for Duplicate names options. If you have Edit Domain Groups and Add Company Layer Domains permissions, these options will also display in the Tools menu. See ***The Tools Menu*** on page 108 for more information on using these options.

Help

The Help menu contains Help, Contextual Help, and About the Dictionary Manager options. See ***Getting Help for the Dictionary Manager*** on page 109 for more information on using these options.

The File Menu

The File menu options include:

Rule

Use this option to create a new Rule, as described in ***Adding and Modifying Rules*** on page 133.

Concept

Use this option to create a new Concept, as described in ***Adding and Modifying Concepts*** on page 143.

Alias List

Use this option to create a new Alias List, as described in ***Adding and Modifying Alias Lists*** on page 147.

Export Domain Group

Use this option to export the Rules and Alias Lists from a specified domain group to a selected location, as described in Exporting a Domain Group.

Exit

Use this option to close the Dictionary Manager windows and end your session.

The Edit Menu

The Edit menu options include:

Find Definitions

Use this option to locate Dictionary objects by specifying relevant attributes, as described in ***Finding Dictionary Definitions*** on page 121.

Concept Tree

Use this option to browse the concept hierarchy, as described in ***Displaying the Concept Tree*** on page 128. You can edit concepts by selecting them directly from the concept tree window. You can add concepts to other dictionary objects by dragging and dropping them directly from the concept tree display.

All Rules

Use this option to display all of the rules defined in the Dictionary, as described in ***The All Rules Window*** on page 110. You can open Rules by double-clicking on a selected rule, or by right-clicking and selecting the Open option. You can delete a rule by right-clicking on the selected rule and selecting the Delete option. You can select multiple rules for open or delete operations.

All Alias Lists

Use this option to display all of the Alias lists defined in the Dictionary, as described in ***The All Alias Lists Window*** on page 111. You can open Alias Lists by double-clicking on a selected list, or by right-clicking and selecting the Open option. You can delete a list by right-clicking on the selected rule and selecting the Delete option. You can select multiple lists for open or delete operations.

The View Menu

The View menu options include:

Rule Creation Preferences

Use this option to specify your preferred rule configuration parameters for the New Rule window. Options include:

- Language patterns tab
- Search Components tab
- Subsume and Supplement settings
- Results Quantity setting

The Dictionary Manager will display the fields for the selected options when you create new Rules. See ***Adding and Modifying Rules*** on page 133 for more information on using these options when creating and modifying Rules.

Cascade All Windows

Use this option to arrange all open windows in the order in which you opened them.

The Tools Menu

The Tools menu options include:

Commit work to central repository

Use this option to add changes you have made in your local copy of the Dictionary to the central repository, as described in ***Committing Changes to the Central Dictionary*** on page 150.

Update from central repository

Use this option to synchronize your local copy of the Dictionary with the central repository, as described in ***Updating a Local Dictionary*** on page 150.

Conflicts

Use this option to check for conflicts in local Dictionary objects, as described in ***Resolving Dictionary Data Conflicts*** on page 151.

Check for duplicate names

Use this option to check for duplicate names in local Dictionary objects, as described in Checking for Duplicate Names.

Edit Domain Groups

Use this option to view, modify, and create domain groups in the central repository, as described in Editing Domain Groups. You must have permission to edit Domain Groups.

Add Company Layer Domains

Use this option to create new Dictionary domains for your application, as described in Creating Domains. You must have permission to add company layer domains.

Getting Help for the Dictionary Manager

The Help menu contains About the Dictionary Manger and Dictionary Manager Help options.

Dictionary Manager Windows

Dictionary Manager windows display lists of objects and details for individual objects. Windows that display lists of objects include:

- ***The Concept Tree Window*** on page 111
- ***The All Rules Window*** on page 110
- ***The All Alias Lists Window*** on page 111
- ***The Domain Groups Window*** on page 120

Windows that display individual objects include:

- ***The Rule Window*** on page 112
- ***The Concept Window*** on page 117
- ***The Alias List Window*** on page 118

Text Highlighting in Dictionary Manager Windows

Dictionary Manager fields that accept IML input use a highlighting scheme to display text. The text highlighting in these fields is designed to help you identify IML elements by applying different colors to the various types of IML elements. Text highlighting is not configurable. The IML highlighting scheme is described in ***Dictionary Manager IML Highlighting Assignments*** on page 109.

Dictionary Manager IML Highlighting Assignments

IML Element	Highlighting Assignment	Example
Infix	dark magenta	OR
Prefix	blue	SENT
Parenthesis	red	( )
Comma	gray	, AND
Enumeration	dark orange	0-2
Literal	dark green	"dog"
Assignment	red	=A, =#B
Keyword	dark cyan	WORD
Label	dark green	#cat
Concept	dark blue	<cat>
Unknown	gold background	
VIL expression	bright cyan background	{VIL_expression}

Find Definitions Window

The Find Definitions window is a tool for locating Dictionary objects. You can use the Find Definitions window to search for Rules, Concepts, and Alias Lists.

You find Dictionary objects by specifying search criteria in the Find Definitions window. The Find Definitions window displays search results as a list of Dictionary objects, sorted in alphabetical order. You can select individual definitions from the list, select multiple items from the list, and sort the definitions by any of the fields in the list.

See ***Finding Dictionary Definitions*** on page 121 for information on specifying search criteria and working with the search results.

The All Rules Window

The All Rules window displays all Rules defined in the Dictionary. The All Rules window lists the following information for each rule from left to right:

Definition Name

This field displays the name of the Rule.

Author

This field displays the name of the user who most recently edited the Rule.

Last edited

This field displays the date that the Rule was last edited. Note that more recent versions of an object may exist in the Central Dictionary, or in other users' local dictionaries.

Domain

This field displays the domain that the Rule is assigned to. A dictionary object is always a member of only one domain.

You can open Rules to view or edit their contents by double clicking, or right-clicking on any of part of a Rule entry and selecting the Open option. You can select multiple rules for the open operation.

The Concept Tree Window

The Concept Tree window displays hierarchies of Concept *type of* or *part of* relationships defined in the Dictionary. The hierarchies originate at specified Concepts, or *Root Concepts* in the application configuration, as described in the *InQuira 6 Application Guide*. The hierarchy displayed below each Root Concept reflects the Concept relationships defined below it in the Ontology.

See ***Concept Relations*** on page 51 for more information on type of and part of Concept relationships.

You can select Concepts from the Concept Tree to display their detail information in the Concept window. See ***The Concept Window*** on page 117 for more information.

The All Alias Lists Window

The All Alias List window displays all Alias Lists defined in the Dictionary. The All Alias List window lists the following information for each rule from left to right:

Definition Name

This field displays the name of the Alias List.

Author

This field displays the name of the user who last edited the Alias List.

Last edited

This field displays the date that the Alias List was last edited. Note that more recent versions of an object may exist in the Central Dictionary, or in other users' local dictionaries.

Domain

This field displays the domain that the Alias List is assigned to. A dictionary object is always a member of only one domain. You can open an Alias list to view or edit its contents by right-clicking on any of part of a rule entry and selecting the Open.

The Rule Window

The Rule window displays all criteria defined for a selected Rule. The Rule window is divided into the following functional areas:

Menu Bar

The Rule window Menu bar contains all of the options available for specifying Rule information. The menu options displayed are determined by your Rule creation preferences and currently specified parameters for the selected Rule.

Selecting some Rule menu options alters the Rule window display by adding tabs or data fields; selecting other options display supplemental windows for specifying Rule information. The Rule menu contains the following options:

- The File menu, which contains the Save, Save As..., and Delete options. See ***The Rule Window File Menu*** on page 113.
- The Edit menu, which contains the Add Business Conditions and Add Answer Action options. See ***The Rule Window Edit Menu*** on page 114.
- The Tools menu, which contains the Concept Tree, Match against a question, Check IML syntax, and View HTML in browser options. See ***The Rule Window Tools Menu*** on page 114.
- The Advanced menu, which contains either Add Question Patterns or Add Question Examples, Add Search Components, Customize Rule precedence, Customize Relative Weights settings, and Customize Maximum Results settings. See ***The Rule Window Advanced Menu*** see "Rules Advanced Menu" on page 114.

Rule information section

The Rule information section lists:

- the author and date of the most recent edit
- the name of the Rule. The name can be any alphanumeric characters. Spaces are allowed.
- optional notes associated with the Rule. Notes can be any alphanumeric characters. Spaces are allowed.

- the domain that the rule is assigned to. For new Rules, the Domain field displays the domain most recently saved to, and only domains for which you have permissions are displayed.

Conditions

The Conditions section contains the sections for specifying the various types of conditions within Rules:

- ***The Question Examples*** see "The Question Examples Tab" on page 115 tab
- ***The Question Patterns tab*** on page 116

A Rule cannot specify both Question Example and Question Pattern conditions. For existing rules, only the tab that contains data will display. For new Rules, the tab display is set by the Rule creation parameters, under View in the Dictionary Manager main window, as described in ***The View Menu*** on page 108.

- the Business Conditions tab displays only if business conditions are specified. See ***The Business Conditions Tab*** on page 116.

Actions

The Actions section contains the following rule components:

- the Maximum results setting displays only if specified for this rule. See Specifying Maximum Results Settings.
- The Answer Action Tab
- The Search Components tab

A Rule cannot specify both Answer actions and Search Component actions. For existing Rules, only the tab that contains data will display. For new Rules, the tab display is set by the Rule creation parameters, under View in the Dictionary Manager main window, as described in ***The View Menu*** on page 108.

The Rule Window File Menu

The Rule window File menu contains the following options:

Save

Use this option to save a new Rule, or to save changes to an existing Rule.

Save As...

Use this option to save the current Rule criteria to a new Rule.

Delete

Use this option to delete a Rule.

The Rule Window Edit Menu

The Rule window Edit menu contains the following options:

Add Business Conditions

Use this option to add business conditions to an Rule. The Business Conditions tab will display in the Conditions section of the Rules window. See ***The Business Conditions Tab*** on page 116.

Add Answer Action

Use this option to an Answer action to an Rule. The Answer tab will display in the Actions section of the Rules window. See The Answer Action Tab.

The Rule Window Tools Menu

The Rules window Tools menu contains the following options:

Show Concept tree

Use this option to display the Concept Tree window, which displays the Concept hierarchy. See The ***Concept Tree WIndow*** see "The Concept Tree Window" on page 111 for more information.

Match Against a question

Use this option to test whether the specified question patterns or question examples match a specified question or request. See ***Testing a Rule Against a Question*** see "Matching a Rule Against a Question" on page 141.

Check IML Syntax

Use this option to check the syntax of all IML expressions within the Rule. See Checking IML Syntax.

View HTML in browser

Use this option to display HTML code specified within the custom text field in a Rules window Answer Section. See Displaying Custom Response HTML.

Rules Advanced Menu

The Rules window Advanced menu contains the following options:

Add Question Patterns

Use this option to add question patterns to the Conditions section of a Rule. Question patterns are symbolic expressions that specify matching criteria for user requests, as described in ***Specifying Question Patterns*** on page 135. You specify question patterns using IML, which is described in ***Working with IML*** see "Working with InQuira Match Language" on page 61.

Add Question Examples

Use this option to add question examples to the Conditions section of a Rule. Question examples are natural language questions that specify matching criteria for user requests, as described in ***Specifying Question Examples*** on page 135.

Add Search Components

Use this option to add search components to the Actions section of a Rule. Search components are the primary method of specifying search criteria for both structured and unstructured data. See ***Search Components*** see "Working with Search Components" on page 44 for more information.

Customize Rule precedence

Use this option to specify precedence for a Rule. You can set Rule precedence to specify Rules that the Rules Engine will not evaluate it this rule is true. See Specifying Rule Precedence.

Customize Subume and Supplement settings

Use this option to specify subsume and supplement settings for a search component in a Rule. Subsume and supplement settings specify how the Rules Engine will process multiple search components that are assigned to the same, or overlapping, parts of the user request. See ***Subsume Settings*** on page 47 and ***Supplement Settings*** on page 48.

Customize Relative Weight settings

Use this option to specify relative weight settings for an Answer Section in a Rule. Relative weight settings specify how the Rules Engine will rank associated factors when scoring results. See Specifying Relative Weight Settings.

Customize Maximum Results settings

Use this option to specify maximum results settings for an Answer section in a Rule. Maximum result settings specify an upper limit for the total number of answers from all specified answer sources that will be displayed. See Specifying Maximum Results Settings.

The Question Examples Tab

The Question Examples tab contains fields for specifying natural language words, phrases, and questions as matching and non-matching criteria, as described in ***Specifying Question Examples*** on page 135.

The Question Patterns Tab

The Question Patterns tab contains a single field for specifying question patterns in the form of IML expressions, as described in ***Specifying Question Patterns*** on page 135.

The Business Conditions Tab

You view, modify and update business conditions using the Business Conditions tab of the Rules window. To display the Business Conditions tab, select Add Business conditions from the Rules window Edit menu. The Business Conditions tab contains the following fields for specifying business conditions:

Condition List

The condition list contains the business conditions that are currently defined for this rule. The conditions list is a display field. You cannot edit this field directly. You can view the individual expressions that define a condition by selecting that condition. You can add conditions using the New button.

Condition Details

The condition detail fields display the conditions, operators, and values that define the condition expressions specified for the selected condition. The condition fields are editable. You can use the condition details fields to modify existing condition expressions and to define new condition expressions. The condition details fields are:

- ***The Condition Field*** on page 116
- ***The Operator Field*** on page 116
- ***The Value Field*** on page 117

The Condition Field

The condition field displays a drop-down list of configured context variables, as described in ***Business Conditions*** see "Working with Business Rules" on page 55. You can select only the variables listed in the drop-down menu.

The Operator Field

The operator field displays a drop-down list of operators that specify a relationship between the specified condition and value. Valid operators are described in ***Business Condition Operators*** on page 58.

You can specify only the operators that appear on the drop-down menu.

The Value Field

The Value field specifies the value for the condition. Valid values for application-specific conditions are dependent on the configured preferences, as described in ***Business Conditions*** see "Working with Business Rules" on page 55.

The Answer Tab

The Search Component Tab

The Content Restriction Tab

The Relative Weights Tab

The Suppress Rules Window

The Concept Window

The Concept window displays all criteria defined for a selected Concept. The Concept window is divided into the following functional areas:

Menu Bar

The Concept window Menu bar contains all of the options available for specifying concept information. Selecting some Concept menu options alters the Concept window display by adding data fields. The Concept menu contains the following options:

- The File menu, which contains the Save and Delete options. See ***Concept File Menu*** see "The Concept Window File Menu" on page 118.
- The Edit menu, which contains the Add New relationship option. See ***Concept Edit Menu*** see "The Concept Window Edit Menu" on page 118.
- Concept information section

The Concept information section lists:

- the author and date of the most recent edit
- the name of the concept. Concept names are divided into three parts, as described in ***Concept Names*** on page 51
- the meaning of the concept. Concept meanings are optional definitions that you specify for reference. They are not used within the Ontology or by the request and response process

- optional notes associated with the concept. Notes can be any number of alphanumeric characters. Spaces are allowed.

See ***Specifying Concept Information*** see "Adding and Modifying Concepts" on page 143 for more information.

Concept Relationship section

The concept relationship section specifies relationships that this concept has with other objects in the Ontology. See ***Specifying Concept Relationships*** on page 144 for more information.

The Concept Window File Menu

The Concept window File menu contains the following options:

Save

Use this option to save changes to a new or existing Concept.

Delete

Use this option to delete a Concept.

The Concept Window Edit Menu

The Concept window Edit menu contains the following options:

Add Concept relationship

Use this option to add relationships to a new or existing Concept. The Concept relationship section will display in the Concept window. See ***Specifying Concept Relationships*** on page 144.

Add Cluster relationship

Use this option to add this Concept to a new or existing Cluster. For new Clusters, the Cluster relationship section will display in the Concept window. For existing Clusters, the Dictionary Manager will prompt you to locate the existing cluster. See ***Specifying Cluster Relationships*** on page 146.

The Alias List Window

The Alias List window displays information for a selected Alias List. The Alias List window is divided into the following functional areas:

Menu Bar

The Alias List window Menu bar contains all of the options available for specifying Alias List information. Selecting some Alias List menu options alters the Alias List window display by adding data fields. The Alias List menu contains the following options:

- The File menu, which contains the Save, Save As..., and Delete options. See ***Alias List File Menu*** see "The Alias List Window File Menu" on page 119.
- The Edit menu, which contains the Alphebetize entries option. See
- The Tools menu, which contains the Import text file option. See ***Alias List Tool Menu*** see "The Alias List Window Tools Menu" on page 120.
- The Advanced menu, which contains the Set overriding option. See ***Alias List Advanced Menu*** see "The Alias List Advanced Menu" on page 120.

Alias List information section

The Alias List information section lists:

- the author and date of the most recent edit
- the name of the Alias List
- optional notes associated with the Alias List. Notes can be any alphanumeric characters. Spaces are allowed.
- the domain that the Alias List is assigned to. For new Alias Lists, the Domain field displays the domain most recently saved to; only domains for which you have permissions are displayed.

See ***Specifying Alias Lists*** see "Adding and Modifying Alias Lists" on page 147 for more information.

Alias List entries

The alias entries are listed in this field. See ***Specifying Alias Lists*** see "Adding and Modifying Alias Lists" on page 147 for more information.

The Alias List Window File Menu

The Alias List window File menu contains the following options:

Save

Use this option to save a new Alias List, or to save changes to an existing Alias List.

Save As...

Use this option to save the current Alias List data to a new Alias List.

Delete

Use this option to delete an Alias List.

The Alias List Window Tools Menu

The Alias List window Tools menu contains the Import text file option. Use this option to import a text file into an Alias List. See ***Importing a Text File into an Alias List*** on page 148.

The Alias List Window Edit Menu

The Alias List window Edit menu contains the Alphabetize entries option. Use this option to alphabetize entries within an Alias List. See ***Alphabetizing an Alias List*** on page 149.

The Alias List Advanced Menu

The Alias List window Advanced menu contains the Set overriding option. Use this option to set overriding options for an Alias List, as described in ***Setting Overriding for Aliases*** on page 147.

The Domain Groups Window

The Domain Groups window lists the currently defined domain groups. Each domain group is represented by an entry in the list. You can select one or more domain groups to view and edit the contents of the group in the Domain Group window.

The Domain Group Window

The Domain Group window contains fields for specifying the domain group name, notes, and member domains. It also lists the available domains that you can specify as members of the group.

The following table describes the domain group fields:

Field	*Description*
Name	This field specifies the domain group name. You can specify any alphanumeric characters.
Notes	This field specifies optional notes about the domain group. You can specify any alphanumeric characters.

Excluded Domains	This field lists the domains that are installed and defined for your application. You can select additional member domains for the current group from this list by dragging and dropping domain names to the Included domains column.
Included Domains	This field lists the domains that are currently specified as members of the domain group. Arrange domain names in order, beginning with the most general and concluding with the most specific. You can arrange domain names by dragging and and dropping names within the Included Domains field. You can remove group members by dragging and dropping domain names to the Excluded domains column.

Including Domains within a Domain List

You include domains in a Domain list by adding their names to the Included Domains field of the Domain Group window. The order that the domain names are listed determines the precedence of the Rules within the application.

To ensure that the most specific Rules are applied within your application, arrange the included domain names in order, beginning with the most general and concluding with the most specific.

Finding Dictionary Definitions

You can use the Find Definitions tool to search for rules, concepts, and alias lists in the Dictionary. You can specify search criteria to limit your search results by definition type, definition name, and specific characteristics of the various types of definitions.

The Find Definitions tool displays search results as a list of definitions, sorted in alphabetical order. Definitions in the list are divided into separate fields that display various types of information. You can select individual definitions from the list, select multiple items from the list, and sort the definitions by any of the fields in the list. See ***Search Results Display*** on page 127 for more information the search results list.

You find Dictionary definitions by:

- opening the Find Definitions window by:
  - selecting Find Definitions from the Edit menu of the Dictionary Manager main window
  - selecting the Find Definitions icon on the Dictionary Manager toolbar
- specifying search criteria

- reviewing the search results

The Dictionary Manager displays the Search Results window.

Specifying Search Criteria

The Find Definitions window displays various fields for specifying search criteria. You can specify general search criteria to search for all types of definitions, or specify definition-specific criteria to search for a single type of definition.

To specify search criteria for all types of definitions, select the All Types radio button. The General search criteria fields display, as described in ***Specifying Search Criteria*** see "Specifying General Criteria" on page 122.

To specify search criteria for a specific type of definition (rule, concept, or alias list) select the corresponding radio button. The General search criteria fields display, and the search fields for the selected definition type display, as described in:

- ***Rule-Specific Criteria*** on page 124
- ***Concept-Specific Criteria*** on page 126
- ***Alias List-Specific Criteria*** on page 126

Specifying General Criteria

You can specify general search criteria for definitions using the following fields:

Field	Description
Name	Use this field to search for definitions by name. You can use the following operators to match specific attributes of the definition name: • contains • exactly matches • starts with • ends with Contains is the default operator.

Notes	Use this field to search for definitions by note text. You can use the following operators to match specific attributes of the note text: • contains • exactly matches • starts with • ends with Contains is the default operator.
Author	Use this field to search for definitions by author. The author is the ID of the user who last edited the definition. You can specify: • any • is one of these, using the Set tool to specify one or more authors Any is the default operator.
Last Edited	Use this field to search for definitions by the date they were last edited. You can use the following operators to match specified attributes of the date: • on any date/none • exactly matches • before this date • after this date On any date/none is the default operator.

Search Criteria Operators

You can use search criteria operators to match specified attributes of your definition search criteria. You can specift string-related, date-related, and domain-related operators, as described in the following table:

String-Related Operators	*Description*
contains	Use this operator to locate items that contain the specified string.
exactly matches	Use this operator to locate items that exactly match the specified string.
starts with	Use this operator to locate items that begin with the specified string.

ends with	Use this operator to locate items that end with the specified string.

Date-related Operators	*Description*
on any date/none	Use this operator to locate items having any or no specified date.
exactly matches	Use this operator to locate items having dates that exactly match the specified date.
before this date	Use this operator to locate items having any date prior to the specified date.
after this date	Use this operator to locate items having any date later than the specified date.

List Operators	*Description*
any	Use this operator to locate items having or belonging to any of the parameters in the associated list.
is one of these	Use this operator to locate items having or belonging to any of the parameters selected from the associated list using the Set tool.

Rule-Specific Criteria

You can specify to search only for rules by selecting the Rules radio button on the Find Definitions window. You can specify general search criteria as described in ***Specifying General Criteria*** on page 122, and Rule-specific criteria, as follows:

Criteria	*Operator*
Domain	Use this field to locate rules within the specified domains. You can use the following operators to specify domain criteria: • any • one of these, using the Set tool to specify one or more domains Any is the default operator.

Question Conditions	Use this field to locate rules with question patterns or example questions that contain the specified string. You can use the following operators to specify question condition criteria: • contains • exactly matches • starts with • ends with Contains is the default operator.
Business Conditions	Use this field to locate rules with business conditions that contain the specified string. You can use the following operators to specify business condition criteria: • contains • exactly matches • starts with • ends with Contains is the default operator.
Purposes	Use this field to locate rules with assigned purposes that match the specified criteria. You can use the following operators to specify purpose criteria: • any • is one of these Any is the default operator.
Methods	Use this field to locate rules with assigned methods that match the specified criteria. You can use the following operators to specify purpose criteria: • any • is one of these Any is the default operator.

Concept-Specific Criteria

You can specify to search only for concepts by selecting the Concepts radio button on the Find Definitions window. You can specify general search criteria as described in ***Specifying General Criteria*** on page 122, and concept-specific criteria, as follows:

Criteria	Operator
Synonym	Use this field to locate concepts with synonyms that match the specified string criteria. You can use the following operators to specify string criteria: • contains • exactly matches • starts with • ends with Contains is the default operator.
Meaning	Use this field to locate concepts with Meaning data that match the specified string criteria. You can use the following operators to specify string criteria: • contains • exactly matches • starts with • ends with Contains is the default operator.

Alias List-Specific Criteria

You can specify to search only for alias lists by selecting the Alias List radio button on the Find Definitions window. You can specify general search criteria as described in ***Specifying General Criteria*** on page 122, and alias list-specific criteria, as follows:

Criteria	*Operator*
Domain	Use this field to locate alias lists within the specified domains. You can use the following operators to specify domain criteria: • any • one of these, using the Set tool to specify one or more domains Any is the default operator.
Key	Use this field to locate alias lists with key entries that include the specified string criteria. You can use the following operators to specify string criteria: • exactly matches • starts with • ends with Exactly matches is the default operator.
Value	Use this field to locate alias lists with value entries that include the specified string criteria. You can use the following operators to specify string criteria: • exactly matches • starts with • ends with Exactly matches is the default operator.

Search Results Display

The Search Results window displays all of the Definitions that matched your search criteria. The results list is divided into columns that contain separate categories of definition information. The results list categories are:

Definition icon

This field displays the Dictionary Manager icon associated with the type of definition.

Definition name

This field displays the definition name, as defined in the information section of the definition.

Author

This field lists the name of the user who last edited the definition.

Last edited

This field lists the date and time of the most recent edit.

Domains

This field displays the domains that the definition is a member of.

The Search Results window lists definitions in alphabetical order by name. You can sort the definitions according to any category by clicking on the column headers. The Search Results window will sort the definitions by the selected category, from least to greatest, or in alphabetical order. You can reverse the sort order by clicking again on the column header.

You can rearrange the order of the Search Results columns by dragging and dropping columns.

The Search Results list will automatically update itself when any of its information changes. For example, if you delete a Concept currently displayed in the Search Results window, it will be deleted from the Search Results list.

You can view or edit definitions in the Search Results list by right-clicking and selecting Open, or by double-clicking. The Dictionary Manager displays the selected definition. To select multiple items from the list, hold down the Control key while you click, or hold down the Shift key while selecting adjacent items. You can display selected items by right clicking and selecting Open, or by right-double-clicking.

You can delete definitions in the Search Results list by right-clicking and selecting Delete. The Dictionary Manager deletes the selected definition. To delete multiple items from the list, hold down the Control key while you click, or hold down the Shift key while selecting adjacent items. You can delete the selected items by right clicking and selecting Delete.

Displaying the Concept Tree

You can display the hierarchy of Concepts in your Dictionary using the Concept Tree window.

To open the Concept Tree window:

- select the Concept Tree icon from the Dictionary Manager main window

Important: You must have a Root Concept specified in your application configuration as described in the InQuira 6 Application Guide.

The Concept Tree hierarchy displays.

You can edit concepts by selecting them directly from the concept tree window. You can add concepts to other dictionary objects by dragging and dropping them directly from the concept tree display.

Chapter 7

Managing User Accounts

The Workbench includes an administrative facility, the User Manager that you can use to view, create, modify, and delete user accounts. To administer Workbench user accounts:

- open the Workbench, as described in the *InQuira 6 Installation Guide*

The Workbench launcher displays.

- select User Manager from the dropdown list of applications.
- Log into the User Manager application

The User accounts window displays.

The InQuira Workbench Users Display

The User accounts window lists the currently defined users for the application.

Each defined user is represented by an entry that is divided into two columns; the User's Full Name column and the Login column. You can use the User accounts window to

- add Workbench users, as described in ***Adding a User*** on page 129
- modify Workbench user information, as described in ***Modifying User Information*** on page 130

Adding a User

You can add users using the Add new user button on the User accounts window.

Note: You must have User Manager permission to add users. For more information, see ***Specifying User Permissions*** on page 131, or consult your system administrator.

To add a user:

- select the Add new user button in the User accounts window

The New user entry window displays.

- complete the user and password information, as described in ***Specifying User Login and Password*** on page 130
- specify permissions information in the New user entry fields, as described in ***Specifying User Permissions*** on page 131

Note: You must specify at least one permission for each user.

- click the Save new user button

The Succesful Save message displays, and the New user entry window closes. The User accounts window lists the new user.

Modifying User Information

You modify user information by selecting a user from the User accounts window.

Note: You must have User Manager authorization to modify user information. For more information, see *InQuira 6 Application Guide* or consult your system administrator.

To modify user information:

- select the desired user from the list in the User accounts window by double-clicking

The Edit user entry window displays the Full Name, Login, and permissions for the selected user.

- modify user and permissions information as described in ***The New User Entry Window*** see "Specifying User Login and Password" on page 130

Note: You must specify at least one permission for each user.

- click the Save Changes button

The Succesful Save message displays, and the Edit user entry window closes.

Specifying User Login and Password

The New user entry window contains the following fields for specifying user and password information:

Field	Description
Full Name	This field specifies the user's full name. It provides a mechanism for associating cryptic login IDs with more descriptive or meaningful names. Specify alphanumeric characters. Spaces are allowed. There is no default.
Login	This field specifies the login ID for the user. This field is required. Specify alphanumeric characters. Spaces are not allowed. There is no default.
New Password	This field specifies the password for the user. This field is required. Specify alphanumeric characters. Spaces are not allowed. When you enter characters in this field, the application displays them as asterisks (*) to ensure security. There is no default. Users with appropriate permissions can change passwords using the process described in ***Modifying User Information*** on page 130.
Confirm new password:	You must re-enter the password specified in the new password field to complete the process of adding a new user.

Specifying User Permissions

The New user entry window contains the following fields for specifying user permissions:

Field	Description
Dictionary Manager	Select this field to specify that the user can access the Dictionary Manager application of the Workbench to create, modify, and delete Dictionary objects specific to your application. You can assign the following advanced Dictionary Manager permissions described below: • Edit Language, Industry Data • Edit Domain Groups • Add Company Layer Domains
User Manager	Select this field to specify that the user can access the Dictionary Manager application of the Workbench to create, modify, and delete users and assign user permissions.
Test Drive	Select this field to specify that the user can access the Dictionary Manager application of the Workbench.

InQuira 6 Analytics	Select this field to specify that the user can access the InQuira 6 Analytics application.
Edit Language, Industry Data	Select this field to specify that the user can create, modify, and delete objects in the industry-specific and base language layers of the Dictionary.
Edit Domain Groups	Select this field to specify that the user can create, modify, and delete members of domain groups within the Dictionary.
Add Company Layer Domains	Select this field to specify that the user can add company layer domains to the Dictionary.
Regression Testing	Select this field to specify that the user can access the Regression testing application within the Dictionary.
InQuira 6 Analytics Administration	Select this field to specify that the user can administer the InQuira 6 Analytics application.

Chapter 8

Customizing the Application Dictionary

You can customize the Dictionary for your application by adding, modifying, and deleting:

- Dictionary Rules as described in ***Adding and Modifying Rules*** on page 133
- Dictionary Concepts and relations as described in ***Adding and Modifying Concepts*** on page 143
- Dictionary Alias Lists as described in ***Adding and Modifying Alias Lists*** on page 147
- Dictionary Domains and Domain Groups as described in Creating Domains and Creating Domain Groups

You customize the Dictionary using the Dictionary Manager application of the Language Workbench. The Language Workbench includes additional functions for:

- testing your Dictionary as described in ***Matching a Rule Against a Question*** on page 141 and ***Testing the Application Dictionary*** on page 153
- updating your local Dictionary from a shared central Dictionary and adding your local changes to the central repository as described in ***Updating Dictionary Data*** on page 149

Adding and Modifying Rules

You can add and modify Rules in the Dictionary using the Dictionary Manager.

You add new Rules using the Rule window, which is available from the Dictionary Manager File menu or tool bar.

You modify existing Rules by:

- locating the desired Rule using:

    - the Find Tool (see ***Finding Dictionary Definitions*** on page 121)
    - the All Rules window (see ***All Rules Window*** see "The All Rules Window" on page 110)

- displaying the Rule window for the desired Rule (see ***Rule Window*** see "The Rule Window" on page 112)

You add and modify Rules by specifying the following required and optional information:

- general Rule information as described in ***Specifying General Rule Information*** on page 134
- language and business conditions as described in ***Specifying Language Conditions*** on page 134 and ***Specifying Business Conditions*** on page 56
- actions as described in Specifying Actions

Specifying General Rule Information

The Dictionary Manager Rule window contains fields for specifying the the following general Rule information.

Name	Use this field to specify the Rule name. The Rule name is required. The Rule name can be any alphanumeric string. Spaces and punctuation are not valid.
Notes	Use this field to enter a free text description of the Rule. This field is optional. You can enter any alphanumeric characters. Spaces and punctuation are valid. The content of the Notes field is for Dictionary Manager users' reference only; it does not affect how the application processes the Rule.
Domain	Use this field to specify the Domain to which this Rule will belong. The pull-down menu displays the available Domains.

Specifying Language Conditions

You specify language conditions in the Conditions section of the Rule window. A Rule can contain only one of the following types of language conditions:

- question examples, as described in ***Specifying Question Examples*** on page 135
- question patterns, as described in ***Specifying Question Patterns*** on page 135

See ***Language Conditions*** on page 31 for more information on how conditions function within Rules.

Specifying Question Examples

You specify question examples using the Question Examples tab in the Conditions section of the Rule window. See ***Question Examples*** on page 32 for more information on how question examples function within Rules.

The Dictionary Manager displays the Question Examples tab by default for new Rules.

To display the Question Examples tab:

- select Add Question Examples from the Advanced menu of the Rule window

Note: You cannot specify question examples and question patterns in the same Rule. If you have previously specified a question pattern, the Dictionary Manager will prompt you to delete question patterns before it displays the Question Example tab.

The Question Examples tab is divided into two columns:

- Questions that should Match
- Questions that should not Match

Each column contains multiple fields. Enter one or more question examples as free text in the appropriate columns. Enter one question per field.

You must specify at least one Question that should Match. You can specify multiple questions to define a range of matching questions. You can limit the range of matching questions by specifying one or more questions that should not match. Questions that should not Match are optional.

You can specify any number of questions in either column. You can add question example fields in either column using the Add line button. You can delete question example fields from either column using the delete [X] icon.

Note: To obtain best results:

- specify the simplest example question possible
- specify match and not match pairs such that the examples are as similar as possible

Specifying Question Patterns

You specify question patterns using the Question Patterns tab in the Conditions section of the Rule window. See ***Question Patterns*** on page 32 for more information on how question patterns function within Rules.

To display the Question Patterns tab:

- select Add Question Patterns from the Advanced menu of the Rule window

Note: You cannot specify question patterns and question examples in the same Rule. If you have previously specified a question example, the Dictionary Manager will prompt you to delete question examples before it displays the Question Patterns tab.

The Question Patterns tab contains a single field. Enter an IML expression, as described in ***Working with InQuira Match Language*** on page 61 to define the desired question pattern.

Specifying Business Conditions

You specify Business Conditions within Rules using the Business Conditions tab in the Dictionary Manager Rule window. See ***Business Conditions*** on page 33 for more information on how Business Conditions function within Rules.

You can specify multiple conditions for a single Rule.

Important: If you specify multiple Business Conditions, the Rules Engine will evalutate the Rule as true only if all of the Business Conditions are true.

You define Business Conditions by creating one or more expressions consisting of conditions, operators, and values as described in ***Business Condition Components*** on page 56.

You can also set the values of configured context variables dynamically based on information within the user request. The Rules Engine can associate a defined context variable with any aspect of a question that can be represented in IML, as described in Setting Question Context Variables.

Viewing Condition Expressions

You can view the condition expressions defined for a selected Business Condition in the Conditions List on the Business Conditions tab. To view condition expressions:

- select the desired condition from the Conditions List

The Condition Details section displays the expressions that define the selected condition.

Adding Business Conditions

You can add Business Conditions to a new or existing Rule by adding the Business Condition tab to the Condition area of the Rule window and specifying one or more condition expressions.

To add a Business Condition:

- select the Add Business Conditions option from the Advanced menu of the Rule window

The Business Condition tab displays in the CONDITION area of the Rule window.

- select the Add new Condition button

The Business Condition window displays.

- specify one or more condition expressions, as described in ***Adding Condition Expressions*** on page 137
- save the Rule to save the specified Business Condition

Important: If you specify multiple Business Conditions, the Rules Engine will evaluate the Rule as true only if all of the conditions are true.

Adding Condition Expressions

To add a condition expression:

- select Add new Condition in the Business Condition tab

The Business Condition window displays.

- specify the desired condition expression using the condition fields. The Business Condition window displays the following fields for specifying condition expressions:

Field	*Description*
Context Variable	This field is a drop-down menu of the available context variables associated with the preferences defined for the application.
Operator	This field is a drop-down menu of the available operators for the selected context variable. The operators are presented in standard English and correspond to the Boolean logic described in ***Business Condition Operators*** on page 58.
Value	This field accepts free text or selected values, such as date and time.

- use the Add a line button to add expression lines within the Business Condition window
- use the delete icon [X] to delete line expression lines within the Business Condition window
- click OK to close the Business Condition window and save the specified condition expressions

Important: If you specify multiple condition expressions, the Rules Engine will evaluate the Business Condition as true if any of the expressions are true.

Specifying Answer Purposes

You specify answer purposes using the Answer tab in the Actions section of the Rule window. See ***Answer Purposes*** on page 35 for more information on how answer purposes function within Rules.

The Dictionary Manager displays the Answer tab by default for new Rules.

To display the Answer tab:

- select Add Answer Action from the Advanced menu of the Rule window

The Answer tab displays a fields for specifying answer parameters, including the Purpose field.

Select the desired answer purpose from the pull-down menu.

Specifying Answer Methods

You specify answer methods using the Answer tab in the Actions section of the Rule window. See ***Answer Methods*** on page 36 for more information on how answer methods function within Rules.

The Dictionary Manager displays the Answer tab by default for new Rules.

To display the Answer tab:

- select Add Answer Action from the Advanced menu of the Rule window

The Answer tab displays a fields for specifying answer parameters, including the Method field. The Dictionary Manager appends the value of the Method field to the Answer tab label.

Select the desired answer method from the pull-down menu. The Answer tab displays additional fields depending on the selected method as described in:

- ***The Custom Content Answer Method*** on page 37
- ***The Specific Excerpt Answer Method*** on page 38
- ***The Search Answer Method*** on page 38
- ***The Structured Query Answer Method*** on page 39
- ***The Glossary Answer Method*** on page 40

Specifying Answer Priorities

You specify answer priorities using the Answer tab in the Actions section of the Rule window. See ***Answer Priorities*** on page 41 for more information on how answer priorities function within Rules.

The Dictionary Manager displays the Answer tab by default for new Rules.

To display the Answer tab:

- select Add Answer Action from the Advanced menu of the Rule window

The Answer tab displays a fields for specifying answer parameters, including the Prioties field.

Select the desired answer priority from the pull-down menu.

Specifying Search Components

You specify Search Components using the Search Component tab in the Actions section of the Rule window. See ***Working with Search Components*** on page 44 for more information on how Search Components function within Rules.

To display the Search Component tab:

- select Add Search Component from the Advanced menu of the Rule window

You can specify mulitple Search Components within a single Rule

The Search Component tab displays fields for specifying the following Search Component parameters:

Field	*Description*
Variable	Enter the variable from the IML expression that you want to associate with the Search Component. See ***Search Component Variables*** on page 45 for more information.
Search Type and Subtype	Select or enter the desired type or subtype as described in ***Search Component Attributes*** on page 45.
Importance	Select the desired Importance. See ***Importance Setting*** on page 49

Subsume and Supplement	Specify Subsume and Supplement settings if desired. You can add Subsume and Supplement fields by selecting the Set Subsume/Supplement option from the Advanced menu. See ***Subsume Settings*** on page 47 and ***Supplement Settings*** on page 48 for more information. **Note:** The default values for Subsume and Supplement are `FALSE`; however, when you select Set Subsume/Supplement from the Advanced menu, the Dictionary Manager sets the values to `TRUE`.
IML/SQL	Enter the IML and/or SQL statements that define the search criteria, using the descending fields to specify the quality of the results as described in ***Quality Levels*** on page 49.

Specifying Relative Weights

You specify relative weights using the Relative Weights tab in the Actions section of the Rule window. See ***Relative Weights*** on page 41 for more information on how relative weights function within Rules.

To display the Relative Weights tab:

- select Add Relative Weights from the Advanced menu of the Rule window

The Relative Weights tab displays a fields for specifying the following weight values:

Field	***Description***
% Search Components	Specify a percentage to weight the Search Component score relative to the Relevance and Recency scores. See ***The Search Component Score*** on page 23 for more information.
% Relevance of Documents	Specify a percentage to weight the Search Component score relative to the Relevance and Recency scores. See ***The Document Relevance and Recency Scores*** on page 24 for more information.
% Recency of Documents	Specify a percentage to weight the Search Component score relative to the Relevance and Recency scores. See ***The Document Relevance and Recency Scores*** on page 24 for more information.

Note: The combined relative weight values must equal 100%.

Specifying Content Restriction

You specify content restriction for Search Actions using the Content Restriction tab in the Actions section of the Rule window. See ***Content Restriction*** on page 42 for more information on how content restriction functions within Rules.

To display the Content Restriction tab:

- select Add Content Restriction from the Advanced menu of the Rule window

The Content Restriction tab displays a fields for specifying the following content restrition criteria:

Field	Description
Collections	The Collections field displays the content collections defined for your application. Select one or more collections.
Attributes	The Attributes field displays the content attributes defined for your application. Select one or more attributes. If you select more than one attribute, specify an AND or OR relationship between the attributes. OR is the default.

Matching a Rule Against a Question

You can test whether a Rule matches a specific question using the Match against a question option on the Rule window Tools menu. The Match against a question function displays information about whether a specific question matches the example questions or question patterns specified within the Rule's condition.

To match a Rule against a question:

- open the Rule in the Dictionary Manager
- select Match against a question from the Tools menu

The Match against a question window displays a question area, a domain group selection field, and a results area.

- select the appropriate Domain Group from the Domain Group dropdown menu. See ***Domain Groups*** on page 30 for more information on defining Domain Groups.
- enter the test question

The results area indicates whether the question matched or failed to match. The content of the results depends on the Rule condition type, which can specify either example questions or question patterns. See ***Language Conditions*** on page 31 for more information on question examples and question patterns within Rule conditions.

Analyzing Question Example Matches on page 142 and ***Analyzing Question Pattern Matches*** on page 142 describe the information that the Match against a question window displays for each condition type.

Analyzing Question Example Matches

When you match a question against a Rule that specifies example question conditions, the Match against a question window displays a message indicating whether or not the test question matched successfully:

```
<question_text> successfully matched.
```

or:

```
<question_text> did not match.
```

If the question matched or failed to match in error, you can use the This is NOT Correct option to add the test question to the list of example questions specified within the Rule.

To add a test question to the set of example questions:

- select This is NOT Correct

The Dictionary Manager adds the test question to the appropriate `should match` or `should not match` category.

Analyzing Question Pattern Matches

When you match a question against a Rule that specifies question pattern conditions, the Match against a question window displays:

- a message indicating whether the test question matched successfully
- the match results table, for matched questions:

Variable	***Match***
This field displays the variables specifed within the question pattern that the Rules Engine assigned as a result of the match. If there are no specified variables, this field displays `Entire Match`.	This field displays the text of the test question. The portion of the question that matched the IML expression is highlighted in red.

You can view the instantiated version of the Rule using the Show Instantiated Rules option. The instantiated Rule displays the values for the instantiated variables that the Rules Engine determined as a result of processing the test question.

To view the instantiated Rule:

- select Show Instantiated Rules

The Rule Instantiations for Matches window displays the Rule in the standard format, with the variables replaced by actual values as instantiated by the Rules Engine.

Adding and Modifying Concepts

You can add and modify Concepts in the Dictionary using the Dictionary Manager.

You add new Concepts using the Concept window, which is available from the Dictionary Manager File menu or tool bar. You modify existing Concepts by:

- locating the desired Concept using:
    - the Find Tool (see ***Finding Dictionary Definitions*** on page 121)
    - the Concept Tree (see ***Displaying the Concept Tree*** on page 128)
- displaying the Concept window for the desired concept (see ***The Concept Window*** on page 117)

You add and modify Concepts by specifying the following required and optional information:

- general Concept information as described in ***Specifying General Concept Information*** on page 143
- Concept relationships as described in ***Specifying Concept Relationships*** on page 144

Specifying General Concept Information

The Dictionary Manager Concept window contains fields for specifying the the following general Concept information.

Name Use this field to specify the Concept name in the format described in

Meaning	Use this field to enter a free text description of the Concept. This field is optional. You can enter any alphanumeric characters. Spaces and punctuation are valid. The content of the Concept Meaning field is for Dictionary Manager users' reference only; it does not affect how the application processes the Concept.
Notes	Use this field to enter a free text description of the Rule. This field is optional. You can enter any alphanumeric characters. Spaces and punctuation are valid. The content of the Notes field is for Dictionary Manager users' reference only; it does not affect how the application processes the Concept.
Priority	Use this field to specify the Concept priority as described in

Specifying Concept Relationships

Concept relationships specify how the Rules Engine will treat the concept in relation to other Concepts in the Dictionary.

You can specify Rules to access concept relationships within both conditions and actions by specifying the ontology traversal functions of variables, as described in ***The Variable Instantiation Language*** see "The Variable Instantiation Language (VIL)" on page 84.

You specify the following types of Concept relationships in the Dictionary using the Dictionary Manager:

- Concept synonyms as described in xref
- optional Concept property relationships according to the Concept's part-of-speech as described in xref and xref
- optional Concept cluster relationships as described in xref

Specifying Concept Synonyms

Synonyms are the words whose meanings you want to specify as contributors to the Concept. When the Rules Engine encounters any of the specified synonyms in a user request or in the application content, it will treat that synonym as an instance of the Concept.

Note: The synonym relationship indicates equivalent terms, not more general (parent) or more specific (child) terms, which are property relationships, as described in ***Concept Part-of-Speech and Relationships*** see "Concept Part-of-Speech and Property Relationships" on page 145.

You specify synonyms as a simple list in the has these synonyms field of the Concept window.

Important: You must specify the Concept headword itself as a synonym of the Concept.

To specify Concept synonyms:

- enter the Concept headword and additional synonym words. Specify each synonym on a separate line.

Concept Part-of-Speech and Property Relationships

Concepts can participate in property relationships. The property relationships that a Concept can participate in are determined by its part-of-speech designation. The part-of-speech designation is a component of the concept name, as described in ***Specifying General Concept Information*** on page 143.

Relationship	Description	Applicable Part-of-Speech
has these types	Specifies more specific concepts that are instances or types of this concept.	• noun • verb • adjective (adj)
is a type of	Specifies more general concepts that this concept is an instance or type of.	• noun • verb
is a part of	Specifies more general concepts that this concept is a component or part of.	• noun • verb • adjective (adj) • adverb (adv)
has these parts	Specifies more specific concepts that are components or parts of this concept.	• noun • verb • adjective (adj) • adverb (adv)
has these adjectives	Specifies adjectives having descriptive properties that apply to this noun concept.	noun

has these nouns	Specifies nouns having descriptive properties that apply to this adjective concept.	adjective (adj)

Specifying Concept Property Relationships see "Concept Part-of-Speech and Property Relationships" on page 145 describes how to specify Concept relationships in the Dictionary Manager Concept window.

Specifying Concept Relationships

To specify a Concept relationship:

- select add Concept relationship from the Concept window Edit menu, or use an empty relationship window
- select the desired relationship from the dropdown menu
- enter the Concept name

You can specify the full Concept name or a partial name if the default Concept name values are correct. Default Concept names are described in ***Specifying General Concept Information*** on page 143. You can also drag and drop a Concept from the Concept Tree window. The Concept Tree window is described in ***The Concept Tree Window*** on page 111.

Specifying Cluster Relationships

Cluster relationships specify a similarity in meaning, rather than equivalence, as in synonym relationships.

To specify a Cluster relationship:

- select add Cluster relationship from the Concept window Edit menu, or use an empty relationship window.
- enter the Concept name

You can specify the full Concept name or a partial name if the default Concept name values are correct. Default Concept names are described in ***Specifying General Concept Information*** on page 143. You can also drag and drop a Concept from the Concept Tree window. The Concept Tree window is described in ***The Concept Tree Window*** on page 111.

Adding and Modifying Alias Lists

You can add and modify Alias Lists in the Dictionary using the Dictionary Manager. You add new Alias Lists using the Alias Lists window, which is available from the Dictionary Manager File menu or tool bar. You modify existing Alias Lists by:

- locating the desired Alias List using the Find Tool, as described in ***Finding Dictionary Definitions*** on page 121 or the All Alias List window, as described in ***The All Alias Lists Window*** on page 111
- displaying the Alias List window for the desired Alias List, as described in ***The Alias List Window*** on page 118

Setting Overriding for Aliases

Alias Lists with the same name can belong to multiple Dictionary Domains used by a single application, and within those lists, duplicate key terms can occur.

If you use multiple Dictionary Domains containing Alias Lists having the same name, definitions in one Alias List will override Definitions having the same key term in the other list. The default override behavior is determinied by the precedence order of the domains specified within the application.

You can specify custom override behavior for alias lists using the Set overriding tool on the Alias List Advanced menu. You can specify either of the following types of overriding:

- entry overriding as described in ***Entry Overriding***
- global overriding as described in ***Global Overriding*** on page 148

Entry Overriding

Entry overriding specifies that entries in the current list will override entries in lists with the same name that exist in higher order domains. Entries in the higher order list that do not have a matching entry in the lower order list will remain active. Entries in the lower order list that do not have a corresponding entry will be active. Matching entries are entries that have identical key terms.

You can configure domain order within an application using the Edit domain groups option of the Dictionary Manager Tools menu.

If an Alias entry belongs to ...	And...	The entry will be...
a lower order list	matches a higher order entry	active
a lower order list	does not match a higher order entry	active
a higher order list	matches a lower order entry	inactive
a higher order list	does not match a lower order entry	active

Global Overriding

Global overriding specifies that a lower order list will override the entire contents of a higher order list, regardless of the entries contained in either list.

If an Alias entry belongs to ...	And...	The entry will be...
a lower order list	matches a higher order entry	active
a lower order list	does not match a higher order entry	active
a higher order list	matches a lower order entry	inactive
a higher order list	does not match a lower order entry	inactive

Importing a Text File into an Alias List

You can import a text file into an Alias List using the Import text file tool on the Alias List Tools menu. The text file must conform to the standard Alias List format as described in ***Working with Alias Lists*** on page 52.

For convenience, you can use tabs as delimitters between key terms and alias terms.

The Import text file tool inserts the contents of the specified text file beginning at the line following the last entry in the list.

To import a text file into an alias list:

- select Tools from the Alias List menu

The Tools pull-down menu displays.

- select Import text file from the Tools pull-down menu

The Import Window displays. The Import Window allows you to locate the desired file by browsing the directories available to you, or by specifying a path.

- locate and select the desired file

The Import text file tool inserts the contents of the specified text file beginning at the line following the last entry in the list.

Alphabetizing an Alias List

You can alphabetize the Alias entries within an Alias List using the Alphabetize entries tool on the Alias List Tools menu. The Alphabetize entries tool alphabetizes the Alias List by key term.

To alphabetize an Alias List:

- select Tools from the Alias List menu

The Tools pull-down menu displays.

- select Alphabetize entries from the Tools menu

The Alphabetize entries tool alphabetizes the Alias List by key term.

Updating Dictionary Data

The application Dictionary resides in a central location, or repository, that can be accessed by multiple users. The Dictionary Manager is designed so that multiple users can make changes to the Dictionary in an orderly fashion.

When you install the Dictionary Manager, it creates a private copy of the Dictionary, usually on your local machine. When you log onto the Dictionary Manager it stores any changes you make to the Dictionary in your local copy. You can add, edit, delete, and test changes to Dictionary data without affecting other users working in their local copies.

Periodically, you need to add your changes to the central Dictionary as described in ***Committing Changes to the Central Dictionary*** on page 150. You also need to incorporate other users' Dictionary changes into your local Dictionary, as described in ***Updating a Local Dictionary*** on page 150.

You can update your local Dictionary without committing changes; however, you cannot commit changes without updating. The Dictionary Manager will inform you if an update is required before committing your changes.

The Dictionary Manager will also test for conflicts between your local Dictionary data and the corresponding data in the central Dictionary. Conflicts can occur if an item that you have made changes to has also been changed in the Central Dictionary since the last time you updated, and your changes to the item are incompatible with the items' current data.

The Dictionary Manager automatically tests for conflicts when you update or commit changes from your local Dictionary, and displays a list of conflicts that you must resolve prior to committing changes.

Committing Changes to the Central Dictionary

To commit changes stored in your local Dictionary to the Central Dictionary, select Commit changes to the Central Dictionary from the Tools menu in the Dictionary Manager main window.

The Dictionary Manager will begin the commit process, and the Commit changes to the Central Dictionary window will display.

During the commit process, the Dictionary Manager automatically:

- checks whether you need to update from the central repository prior to committing changes
- updates each changed item in your local Dictionary
- tests it for conflicts

The Dictionary Manager will not commit changes if it detects a conflict. See ***Resolving Dictionary Data Conflicts*** on page 151. The Dictionary Manager will not perform any other functions during the commit operation, since doing so could affect the data being transferred.

When the commit operation is complete, use the Close button to close the Commit changes to the Central Dictionary window.

Updating a Local Dictionary

To update your local Dictionary with changes that other users have committed to the Central Dictionary, select Update from the Central Dictionary from the Tools menu in the Dictionary Manager main window.

The Dictionary Manager will begin the update process, and the Updating from Central Dictionary window will display.

During the update process, the Dictionary Manager automatically tests for conflicts and updates your local Dictionary with any items in the Central Dictionary that have changed since your last update.

The Dictionary Manager will not commit changes if it detects a conflict. See ***Resolving Dictionary Data Conflicts*** on page 151. The Dictionary Manager will not perform any other functions during the update operation, since doing so could affect the data being transferred.

When the update operation is complete, use the Close button to close the Update from the Central Dictionary window.

Resolving Dictionary Data Conflicts

The Dictionary Manager will not commit changes if it detects a conflict. To proceed with the updating or committing changes, you must resolve the conflict.

To display the Conflict list:

- select Conflicts from the Dictionary Manager main window

The Conflicts window displays a list of current data conflicts.

To resolve Dictionary data conflicts:

- select an item from the conflicts list

The conflict resolution interface displays.

Combining Changes to Resolve Dictionary Conflicts

If you and another user have both edited a Dictionary object, you may combine your changes. The interface for this situation allows you to copy any of the other user's changes into your copy and then save the result. You can also choose one of the Definitions over the other. This display highlights the section of the other user's Dictionary object that is different from yours.

Chapter 9

Testing the Application Dictionary

You can test the application Dictionary using the Workbench's Test Drive application, which provides detailed information about the Dictionary Rules, language analysis processing, unstructured and structured information retrieval, application messages, and XML (GIML) associated with the response to an individual question.

You use the Test Drive by:

- logging onto the application within the Workbench as described in ***Accessing the Test Drive Application*** on page 153
- setting test drive parameters as described in ***Setting Test Drive Parameters*** on page 154
- entering individual test questions as described in ***Entering a Test Question*** on page 155
- analyzing the results as described in ***Analyzing Test Results*** on page 156

Accessing the Test Drive Application

You can access Test Drive:

- from the Workbench Launcher
- from other Workbench applications, such as the User Manager or Dictionary Manager

To access Test Drive from the Workbench Launcher:

- enter a valid User name and password
- select Test Drive from the Application dropdown menu

To access Test Drive from other Workbench applications:

- double-click on the Test Drive icon, displayed in the lower right corner of the main application window

The Test Drive Main Window displays.

The Test Drive Main Window

The main window provides access to all of the functions of the Test Drive application:

- the question area accepts individual questions as input, as described in ***Entering a Test Question*** on page 155
- the test parameters allow you to select the Dictionary domain group that you want to test questions against, and whether you want to test changes to Concept information prior to re-indexing the application content, as described in ***Setting Test Drive Parameters*** on page 154
- the results area provides tabs that you can select to view several types of information about the application's response to the test question, as described in ***Analyzing Test Results*** on page 156

Setting Test Drive Parameters

You can set the following parameters for the Test Drive application in the main window:

- the Dictionary domain group to test question against, as described in ***Specifying the Test Domain Group*** on page 154
- whether to test Concept information changes prior to re-indexing application content, as described in ***Testing Concept Information Prior to Reprocessing Content*** on page 154

Specifying the Test Domain Group

The Test Drive main window displays the domain groups that are defined within the application configuration. You can select any of the currently defined domain groups. The selected domain group determines the set of Rules, Concepts, and Alias Lists that the application will use when testing questions.

See ***Domain Groups*** on page 30 for more information.

Testing Concept Information Prior to

Reprocessing Content

The Test Drive main window contains a field to specify Concept expansion. Concept expansion is a simplified means of simulating the effect of Concept changes on the application. It enables you to test Concept-related changes to the Dictionary prior to re-processing the application content.

The Language Analyzer uses Concept definitions to establish semantic relationships for words and phrases occurring in the application content, and records that information in the index during content processing. The process of recording semantic relationship information in the index is called annotation.

The Concept expansion function creates a temporary layer that records the recently updated Concept information in the Dictionary in an expanded IML format. The Test Drive application uses this layer when evaluating test questions, thereby simulating the effect that the updated Concept information will have when you re-process the application content.

For example, consider a concept `<noun.artifact.car>`, having the synonyms `auto`, `automobile`, `car`, and `vehicle`.

Concept expansion creates an IML expression (`auto OR automobile OR car OR vehicle`) for use whenever the concept `<noun.artifact.car>` is invoked. The IML expression simulates the updated synonym relations that will take effect the next time the Language Analyzer annotates the application content during content processing.

Important: Concept expansion produces a potentially larger set of Concept matches than the normal annotation process; therefore, it produces results that may differ significantly from those produced by the application.

Concept expansion matches all of the synonyms in the Dictionary for the Concepts occurring in the question; however, the application matches synonyms only for those Concepts that are annotated in the application content. Concepts that the Language Analyzer does not annotate include:

- ambiguous base language Concepts where there is no specified preferred meaning, or sense
- Concepts that form a part of a larger concept with a more specific meaning, for example `brake` within `brake pad`

Entering a Test Question

The question area of the main window is functionally identical to the question area of the User Interface. You use the Test Drive application by entering individual questions. Test Drive displays the results data in the results area, as described in ***Analyzing Test Results*** on page 156.

Note: Test Drive also displays the response page for each test question in a separate browser window, formatted using the default User Interface.

Analyzing Test Results

The Test Drive results area displays a various data associated with the application's response to the test question. The results area is divided into the following tabbed sections that you can select:

Tab	Description
Rules Engine	This section displays the Rules that the Rules Engine matched when processing the question, as described in ***Analyzing Matched Rules*** on page 156.
Annotations	This section displays the language analysis annotations associated with the tokens that make up the question, as described in ***Analyzing Annotations*** on page 157.
Unstructured Results	This section displays the Search Components, index result, excerpt, and related information for each answer from unstructured content, as described in ***The Unstructured Results Tab*** on page 163.
Structured Retrieval	This section displays the SQL, IML, new SQL, and removed and changed fields for each answer from structured content, as described in ***The Structured Results Tab*** on page 165.
All Messages	This section displays the all messages associated with the question that were written to the application log file, as described in ***The All Messages Tab*** on page 166.
XML Results	This section displays the GIML that the application created to generate the response page and its associated metadata for this question, as described in ***The XML Results Tab*** on page 166.

Analyzing Matched Rules

The Rules Engine tab displays information about the Dictionary Rules that matched the test question. The Rules Engine tab is divided into the following fields:

Field	Description
Rule Name	This field lists the names of the Rules within the selected domain group that the Rules Engine matched to the test question. The Rules are listed in the order in which they were matched.
Domain	This field lists the domain to which each matched Rule is assigned.

You can view any Rule by selecting (double-clicking) it from the Rule Name column. The Dictionary Manager application opens the selected Rule. ***Dictionary Rules*** on page 15 provides an overview of Rules and their function within the application.

Analyzing Annotations

The Annotations tab displays the results of the Language Analyzer's processing of the test question. Annotations are various types of data that the application associates with elements of the question. The Language Analyzer and the Rules Engine add annotations during analysis that determine the interpretation and response to the request. The types of annotations that result from language analysis are described in ***Annotation Types*** on page 158.

When you select the Annotations tab, the Annotations display area is empty until you select an Annotation Type.

- select an Annotation Type by clicking on a tab:

The annotation display area displays:

- the Annotation Location field
- the default annotations summary display

You can use the Annotation Location field to restrict the annotations in the display area, as described in ***Restricting the Annotation Display*** on page 158.

Selecting Annotation Locations and Types

The Annotations tab displays each question word as a selectable button in the Annotation Location field, and each Annotation Type as a selectable tab to the left of the display area.

You can view Annotation Type data for all applicable question words, or for a selected question word. The Annotation tab displays data for all applicable question words by default. When you select an Annotation Type, the question words for which that type of data exists are highlighted in red in the Annotation Location field.

To display Annotation Type data for an individual question word:

- select the desired word in the Annotation Location field

The selected word will display be highlighted in red.

- select the desired Annotation Type tab, as described in ***Annotation Types*** on page 158

Viewing Annotations for Selected Question

Words

The Annotation Location field displays when you select an Annotation Type tab. It displays each question word as a selectable button. The default Annotation Location display highlights the question words that correspond to the first data entry for the selected annotation type.

You can display Annotation Type data as:

- a summary display of all question data, which is the default
- a restricted display of data for a selected question word, as described in ***Restricting the Annotation Display*** on page 158

Note: Once you restrict the display be selecting a question word, you cannot re-display the summary display.

Restricting the Annotation Display

You can restrict the Annotation Type display by selecting a question word from the Annotation Location.

To restrict the Annotation Type display to a single question word:

- select the desired word by clicking a button in the Annotation Location field

The Annotation Type display shows only entries associated with the selected question word.

You can view data for other question words by selecting them from Annotation Location field. You can view other types of annotations by selecting them from the Annotation Type tabs, as described in ***Selecting Annotation Locations and Types*** on page 157.

Annotation Types

The Annotation Type tab displays information about the following annotation types:

Annotation Type	Description
Ambiguous	This tab displays information about question words that matched multiple Concepts, and for which no prefered sense is specified in the Dictionary. Ambiguous annotations are in the form: `<Concept:(n):mmmmmm:pos.domain.headword>` You can view detailed information about any Concept associated with the ambiguous question word by selecting (double-clicking) it. The Dictionary Manager will display the Concept window for the selected concept.

Concept Synonyms	This tab displays the defined Dictionary synonyms for the selected question words. Synonym annotations are in the form: `<Concept:(nn):mmmmmm:pos.domain.headword>` You can view detailed information about any synonym in the display by selecting (double-clicking) it. The Dictionary Manager will display the Concept window for the selected concept. See ***Concept Synonyms*** on page 51 for more information.
Label	This tab displays information about labels assigned to each question word by the language analysis process and the Rules Engine. The Language Analyzer assigns labels as described in ***Language Analysis Labels*** on page 160 The Rules Engine assigns labels in the form of variables, as described in ***Variables*** on page 82. Labels can span multiple question words. If you select a label that spans multiple words, all of the words will be highlighted. If you select a question word in the Annotation Location field that is within the span of a currently selected label, the selected word will be highlighted against a white background.
No Concept Identified	This tab displays a value of `TRUE` if a selected question word is not recognized as a concept in the Dictionary. This is not necesarily cause for action; for example, words that are labeled as skip words will receive a value of `true`.
Part of Speech	This tab displays part of speech information for the selected question word. ***Part of Speech Annotations*** on page 160 describes the designated parts of speech.
Phrase	This tab displays phrase typing information for the selected question word. ***Phrase Annotations*** on page 161 describes the designated phrase types.
Search Component	This tab displays the search components that were assigned to the selected question word or range of question words. If you select a search component that spans multiple words, all of the words will be highlighted. If you select a question word in the Annotation Location field that is within the span of a currently selected search component, the selected word will be highlighted against a white background.
Stem	This tab displays the stem, or basic form of the selected question word.

Language Analysis Labels

The Language Analyzer assigns various types of labels, which are distinct from the variable-related labels that the Rules Engine assigns. Language Analyzer-assigned lables are generally notated in all capital letters. The Language Analyzer assigns the following types of labels:

- part of speech and bracketing labels. These labels use the part of speech tags described in ***Part of Speech Annotations*** on page 160.
- orthographic labels, which include PUNCTUATION, CAPITALIZED, ALL_CAPS and NUMBER
- synset labels, which include FIRST_CONCEPT and CONCEPT
- scoring labels, which include BASE_LEVEL, INDUSTRY_LEVEL and CUSTOMER_LEVEL. Scoring labels correspond to the synset score assigned in the Dictionary.
- the ANAPHORA label, which is added for concepts added during anaphora resolution

Part of Speech Annotations

The Language Analyzer identifies and annotates the part of speech of words within both questions and indexed content using the tags described in the following table.

Note: The annotations and descriptions are not configurable; they are published for reference only.

Annotation	*Description*
CC	Coordinating conjunction
CD	Cardinal number
DT	Determiner
EX	Existential there
FW	Foreign word
IN	Preposition or subordinating conjunction
JJ	Adjective
JJR	Adjective, comparative
JJS	Adjective, superlative
LS	List item marker

MD	Modal
NN	Noun, singular or mass
NNS	Noun, plural
NNP	Proper noun, singular
NNPS	Proper noun, plural
PDT	Predeterminer
POS	Possessive ending
PRP	Personal pronoun
PRP$	Possessive pronoun (prolog version PRP-S)
RB	Adverb
RBR	Adverb, comparative
RBS	Adverb, superlative
RP	Particle
SYM	Symbol
TO	to
UH	Interjection
VB	Verb, base form
VBD	Verb, past tense
VBG	Verb, gerund or present participle
VBN	Verb, past participle
VBP	Verb, non-3rd person singular present
VBZ	Verb, 3rd person singular present
WDT	Wh-determiner
WP	Wh-pronoun
WP$	Possessive wh-pronoun (prolog version WP-S)
WRB	Wh-adverb

Phrase Annotations

The Language Analyzer identifies and annotates phrases within the questions and the indexed content using the tags described in the following table.

Note: The annotations and descriptions are not configurable; they are published for reference only.

Annotation	Description
ADJP	Adjective Phrase
ADVP	Adverb Phrase
CONJP	Conjunction Phrase
FRAG	Fragment
INTJ	Interjection. Corresponds approximately to the part-of-speech tag UH.
LST	List marker. Includes surrounding punctuation.
NAC	Not a Constituent; used to show the scope of certain prenominal modifiers within an NP.
NP	Noun Phrase
NX	Used within certain complex NPs to mark the head of the NP. Corresponds very roughly to N-bar level but used quite differently.
PP	Prepositional Phrase
PRN	Parenthetical
PRT	Particle. Category for words that should be tagged RP.
QP	Quantifier Phrase (i.e. complex measure/amount phrase); used within NP.
RRC	Reduced Relative Clause
UCP	Unlike Coordinated Phrase
VP	Verb Phrase
WHADJP	Wh-adjective Phrase. Adjectival phrase containing a wh-adverb, as in how hot.
WHAVP	Wh-adverb Phrase. Introduces a clause with an NP gap. May be null (containing the 0 complementizer) or lexical, containing a wh-adverb such as how or why.
WHNP	Wh-noun Phrase. Introduces a clause with an NP gap. May be null (containing the 0 complementizer) or lexical, containing some wh-word, e.g. who, which book, whose daughter, none of which, or how many leopards.
WHPP	Wh-prepositional Phrase. Prepositional phrase containing a wh-noun phrase (such as of which or by whose authority) that either introduces a PP gap or is contained by a WHNP.

X	Unknown, uncertain, or unbracketable. X is often used for bracketing typographical errors and in bracketing the...the-constructions.

The Unstructured Results Tab

The Unstructured Results tab displays information about the unstructured information retrieval results for the test question. The Unstructured Results tab displays answers within numbered tabs, arranged vertically in the order in which they were scored by the Evaluator and displayed within the User Interface. See ***Scoring and Ranking Responses*** on page 21 for more information on scoring.

To view information for an answer:

- select the corresponding numbered tab

The Unstructured Results tab displays the following information for each answer.

- the search components used to locate the answer (as well as the search components that were assigned and not used), as described in ***Search Component Information*** on page 163
- the indexed content associated with the answer as described in ***Index Result Information*** on page 164
- the excerpt created from the indexed content as described in ***Excerpt Result Information*** on page 164
- related Rule, collection, document type, answer scope, and scoring information, as described in ***Related Unstructured Information*** on page 164

Search Component Information

The Search Component field of the Unstructured Results tab displays the Search Components associated with the selected answer.

You can select the Show Unused field to display all of the Search Components that the Rules Engine assigned to the question, or display only the Search Components that the Rules Engine applied to the corresponding results from the indexed content. See ***Working with Search Components*** on page 44 for more information on Search Components and how the Rules Engine assigns them.

You can view detailed information for a Search Component in a separate window.

To view detailed Search Component information:

- select the Search Component by double-clicking the item in the Search Component field

The Instantiated Search Component window displays.

The Instantiated Search Component window displays the relevant elements of the question as they apply to the Search Component parameters defined in the associated Rule. You can view the Rule that defines the selected Search Component using the View Rule button of the Instantiated Search Component window.

To view the Rule that defines the selected Search Component:

- select the View Rule button on the Instantiated Search Component window

The Dictionary Manager displays the corresponding Rule.

Index Result Information

The Index Result field of the Unstructured Results tab displays the matched index content associated with the selected answer. The portion of the matched index content that reflects the scope of the currently selected search component is displayed in highlighted text.

Excerpt Result Information

The Excerpt Result field of the Unstructured Results tab displays the text that the Excerpt Service compiled for presentation in the User Interface. The Excerpt Service creates excerpts dynamically by applying its algorithms to the high-scoring Search Components to compile and format relevant heading and body text for answer presentation.

The Excerpt Result field displays the Index Result text highlighted in blue, and the scope of the matched Search Component highlighted in red.

Related Unstructured Information

The Related Information area of the Unstructured Results tab displays the following information:

Field	*Description*
From Rule	This field displays the name of the Rule that retrieved the selected answer.

Collection name	This field displays the name of the document collection in which the answer is located. Document collections are defined within the content acquisition process, as described in the *InQuira 6 Application Guide*.
ID	This field displays the ID number assigned to the answer source document duing the content acquisition process.
Document type	This field displays the file type of the answer document, such as HTML or PDF.
Scope type	This field displays the scope type associated with the matched answer. Scope types include `sentence`, `section`, and `document`.
Range	This field displays the offsets, which indicate the positions of tokens (words) within the index, associated with the beginning and end of the Search Component scope.
Overall Score	This field displays the overall score, which is calculated using weighted averaging and normalization of the component search, recency, and relevance scores.
Search Score	This field displays the search score.
Recency Score	This field displays the document recency score.
Relevancy Score	This field displays the document relevance score.
Document Ranking	This field displays a weighted average of links to the answer source document.

See ***Scoring and Ranking Responses*** on page 21 for more information about scoring.

The Structured Results Tab

The Structured Results tab displays the following information about the structured information retrieval results for the test question:

- the SQL statement that was generated by the Rule that matched the question
- any IML generated by the matching Rule
- any new SQL statement resulting from removed or changed fields
- any removed fields

- any changed fields

The All Messages Tab

The All Messages tab displays all messages written to the application's log files associated with the test question.

The XML Results Tab

The XML Results tab displays the GIML elements that make up the application response, and the element's attributes and their values. The GIML elements are displayed as a tree in the Structure column of the display, and their corresponding attributes are displayed in the Name and Value columns of the Attributes area.

You can select element nodes within the Structure column to collapse and expand the tree. The Attributes area displays the attribute names and values for the selected element.

The Message Element

The Message element is the parent element in the GIML results. The Message element has a single attribute, `type`, which has a single possible value, `response`.

Significant child elements of the Message element include:

- ***Parameter Elements*** on page 166
- ***The Responses Element*** on page 168
- ***The Query Element*** on page 171

Parameter Elements

The XML Results tab displays various parameter elements associated with the request and the response.

Request-related parameters include:

Name	Value
Question	This value is the text of the request.
TransactionID	This value is the unique ID assigned to the request.
PriorTransactionID	This value is the unique ID assigned to the previous request within the user session.

Additional parameters include ***User Agent Parameters*** on page 167 and ***Answer Purpose Parameters*** on page 167.

User Agent Parameters

The user agent parameters contain information related to the HTTP message associated with the request. User agent parameters can convey HTTP header information such as host, connection, and cookie values.

Answer Purpose Parameters

The Answer Purpose parameters contain information about the number of answers for each answer purpose. The User Interface uses this information to format the results pages.

Name	***Value***
page_size	This is the value for the number of answers of this purpose to display per results page, as set in the application configuration.
page_start	This value indicates the starting position of the current page for answers of this purpose. For example, the first group of answers of the same purpose will have a `page_start` value of `0`. (The number of answers grouped on the first results page is determined by the page_size value for that answer purpose.) If the `page_size` value is `5`, then the `page_start` value for the next results page would be `5`.
page_more	This value indicates the number of answers to be displayed after the current page. For example, if there are `50` answers, and each results page displays `10`, the `page_more` value for the first results page will be `50`, and the `page_more` value for the second results page will be `40`.

The standard answer purposes defined in the application include:

- FEATURE_CONTENT
- ANSWER
- DEFINE
- CONVERSE
- JUMP_TO_PAGE
- PROMOTE

- ACT
- LINK_TO_CATEGORY
- CLARIFY

See InQuira 6 User Interface Guide and InQuira 6 Application Guide for more information on defining and using answer purposes.

The Responses Element

The Responses element is a child of the Message element. The Responses element has a single attribute, `type`, which has a single possible value, `response`.

The child element of the Responses element is the Answer element, as described ***The Answer Element*** on page 168.

The Answer Element

The Answer element describes the contents and metadata associated with individual answers returned by the unstructured information retrieval process. The Answer element has the following attributes:

Name	*Value*
collectionId	The value is the internally defined document collection ID of the collection that contains the answer document. See the *InQuira 6 Application Guide* for more information on defining document collections.
collectionName	This value is the document collection name, as assigned when the collection was defined. See the *InQuira 6 Application Guide* for more information on defining document collections.
docid	The value is the internally defined document ID. Document IDs are assigned during content processing.
doctype	The value is the document type, for example, `HTML` or `PDF`, as identified during content processing.
score	The value is the overall score for this answer.

similar_count	The value is the number of similar answers, as determined internally by the similar answer function, which groups answers having similar characteristics, and provides access to them through the similar answers link, thereby reserving the answer area for more distinct answers. The similar answers function is not configurable.
type	The value is the type of information retrieval that generated the answer, for example, `Unstructured` indicates that the unstructured information retrieval process generated the answer.

The Answer element has the following child elements that describe the text content of the answer:

- ***The Section Element*** on page 169
- ***The Title Element*** on page 169
- ***The Text Element*** on page 170

The Section Element

The Section element is a child of the Answer element. Section elements correspond to the sections within the source document as identified during content processing. See InQuira 6 Application Guide for more information on content processing. The Section element has no attributes.

Section elements can contains multiple Title and Text elements, as described in ***The Title Element*** on page 169 and ***The Text Element*** on page 170, and can also be contained within Text elements.

The Title Element

The Title element is a child of the Answer element. Title elements correspond to the titles identified within the source document during content processing. The Title element has the following attributes.

Name	***Value***
idx	This value is the internal identifier of the index that contains the answer document.
url	This value is the URL of the answer document.

The Text Element

The Text element is a child of the Answer element. Text elements correspond to the paragraphs identified within the source document during content processing. The Text element has the following attributes.

Name	Value
idx	This value is the internal identifier of the index that contains the answer document.
url	This value is the URL of the answer document.

The Snippet Element

The Snippet element is a child of the Text element. Snippet elements are segments of the excerpt compiled for the answer. Each excerpt is divided into multiple Snippet elements, and each snippet has an attribute, `lvl`, the value of which indicates the degree of matching associated with the snippet.

Name	Description
lvl	This value corresponds to the degree of of matching associated with the snippet. Values include: `0 -` this snippet contains no matches, only context information `1 -` this snippet contains no primary or secondary matches `2 -` this snippet contains secondary matches, for example, synonym or morphological variants `3 -` this snippet contains direct matches

The Timestamp Element

The Timestamp element contains date and time information that indicates the most recent update of the answer document within the Content Store. The Timestamp element contains child elements that correspond to the date, month, year, hour, minute, second, and millisecond of the document's timestamp.

The Query Element

The Query element contains the Question element, as described in ***The Question Element*** on page 171, and has no attributes.

The Question Element

The Question element contains information about the question being processed. The Question element contains the Original element, as described in ***The Original Element*** on page 171, and the Paraphrase element, as described in ***The Paraphrase Element*** on page 171. The Question element has one attribute, `TransactionID`, which is the unique ID assigned to the request.

The Original Element

The Original element contains the original request as entered by the user. The Original element has no attributes.

The Paraphrase Element

The Paraphrase element contains the revised request as rendered and processed by the User Interface. See *InQuira 6 User Interface Guide* for more information on the paraphrase function of the User Interface. The Paraphrase element has no attributes.