BEA Liquid Data for WebLogic™

Product Overview

Release: 1.0
Document Date: October 2002
Revised:

Copyright

Restricted Rights Legend

Trademarks or Service Marks

Product Overview

Part Number	Date	Software Version
886-002002-03	October 2002	1.0

Contents

3. Liquid Data Concepts

4. Liquid Data Quick Start

Index

About This Document

This document provides an introductory overview of the BEA Liquid Data for WebLogic™ product. This document covers the following topics:

- Chapter 1, "Introducing Liquid Data," provides a general introduction to Liquid Data, including a summary of key capabilities and user roles associated with a Liquid Data implementation.
- Chapter 2, "Liquid Data Architecture," provides an overview of the Liquid Data architecture and product components.
- Chapter 3, "Liquid Data Concepts," introduces concepts that you should understand in order to use Liquid Data effectively.
- Chapter 4, "Liquid Data Quick Start," provides an overview of the key tasks to start using Liquid Data for data access and aggregation.

What You Need to Know

This document provides a general introduction to the BEA BEA Liquid Data for WebLogic product. You should read it to learn the basics of using Liquid Data.

This document does not require any particular expertise, but a familiarity with the following topics helps: data integration and aggregation concepts, business and technical knowledge about the environment in which you will install and use Liquid Data, the BEA WebLogic® Platform™, and query concepts (including the XQuery standard).

e-docs Web Site

BEA product documentation is available on the BEA corporate Web site. From the BEA Home page, click on Product Documentation or go directly to the "e-docs" Product Documentation page at http://e-docs.bea.com.

How to Print the Document

You can print a copy of this document from a Web browser, one file at a time, by using the File—>Print option on your Web browser.

A PDF version of this document is available on the Liquid Data documentation Home page on the e-docs Web site (and also on the documentation CD). You can open the PDF in Adobe Acrobat Reader and print the entire document (or a portion of it) in book format. To access the PDFs, open the Liquid Data documentation Home page, click the PDF files button and select the document you want to print.

If you do not have the Adobe Acrobat Reader, you can get it for free from the Adobe Web site at http://www.adobe.com/.

Related Information

For more information in general about Java and XQuery, refer to the following sources.

- The Sun Microsystems, Inc. Java site at:

  `http://java.sun.com/`

- The World Wide Web Consortium XML Query section at:

  `http://www.w3.org/XML/Query`

For more information about BEA products, refer to the BEA documentation site at:

`http://edocs.bea.com/`

Contact Us!

Your feedback on the BEA Liquid Data documentation is important to us. Send us e-mail at **docsupport@bea.com** if you have questions or comments. Your comments will be reviewed directly by the BEA professionals who create and update the Liquid Data documentation.

In your e-mail message, please indicate that you are using the documentation for the BEA Liquid Data for WebLogic 1.0 release.

If you have any questions about this version of Liquid Data, or if you have problems installing and running Liquid Data, contact BEA Customer Support through BEA WebSupport at **www.bea.com**. You can also contact Customer Support by using the contact information provided on the Customer Support Card, which is included in the product package.

When contacting Customer Support, be prepared to provide the following information:

- Your name, e-mail address, phone number, and fax number
- Your company name and company address
- Your machine type and authorization codes
- The name and version of the product you are using
- A description of the problem and the content of pertinent error messages

Documentation Conventions

The following documentation conventions are used throughout this document.

Convention	Item
boldface text	Indicates terms defined in the glossary.
Ctrl+Tab	Indicates that you must press two or more keys simultaneously.
italics	Indicates emphasis or book titles.

`monospace text`	Indicates code samples, commands and their options, data structures and their members, data types, directories, and file names and their extensions. Monospace text also indicates text that you must enter from the keyboard.
	Examples:
	`#include <iostream.h> void main ( ) the pointer psz`
	`chmod u+w *`
	`\tux\data\ap`
	`.doc`
	`tux.doc`
	`BITMAP`
	`float`

`monospace boldface text`	Identifies significant words in code. *Example*: `void` **`commit`** `( )`
`monospace italic text`	Identifies variables in code. *Example*: `String` *`expr`*
UPPERCASE TEXT	Indicates device names, environment variables, and logical operators. *Example*s: LPT1 SIGNON OR

Convention	Item
`{ }`	Indicates a set of choices in a syntax line. The braces themselves should never be typed.
`[ ]`	Indicates optional items in a syntax line. The brackets themselves should never be typed. *Example*: `buildobjclient [-v] [-o name ] [-f file-list]... [-l file-list]...`
`|`	Separates mutually exclusive choices in a syntax line. The symbol itself should never be typed.
`...`	Indicates one of the following in a command line: ■ That an argument can be repeated several times in a command line ■ That the statement omits additional optional arguments ■ That you can enter additional parameters, values, or other information The ellipsis itself should never be typed. *Example*: `buildobjclient [-v] [-o name ] [-f file-list]... [-l file-list]...`
. . .	Indicates the omission of items from a code example or from a syntax line. The vertical ellipsis itself should never be typed.

CHAPTER

1 Introducing Liquid Data

This topic provides a high level overview of the BEA Liquid Data for WebLogic™ product. It contains the following sections:

- What is BEA Liquid Data for WebLogic?
- Liquid Data Capabilities
- Liquid Data Roles and Responsibilities

What is BEA Liquid Data for WebLogic?

BEA Liquid Data for WebLogic is a data access and aggregation product for *Information Visibility*, allowing a real-time unified view of disparate enterprise data. Information visibility is essential to enable greater supply-chain visibility, a unified view of customer information, and better decision-making.

BEA Liquid Data for WebLogic provides a cost effective, standard way to rapidly aggregate and expose logical views from any number of heterogeneous sources (including Web services, databases, flat files, XML files, applications and Web sites). This enables developers to re-use information across applications without dealing with the complexity of the underlying data.

One of the greatest challenges for IT organizations is that today's technologies do not adequately address the complexity inherent in aggregating data from distributed systems. BEA Liquid Data for WebLogic allows IT departments to easily aggregate data, in real time, from many sources within and outside of the enterprise, and tailor it for different business users, providing information visibility anywhere in the enterprise.

BEA Liquid Data for WebLogic is an easy-to-use product that simplifies the development of applications and portals that require aggregated, real-time information from disparate or heterogeneous sources. BEA Liquid Data for WebLogic returns results in XML for easy consumption by Web applications.

BEA Liquid Data for WebLogic delivers flexible, reusable information without the high cost and time consuming custom coding of queries and data transformations required with other products and solutions. Developers access BEA Liquid Data for WebLogic as simple Business Information Services created and published by a company's domain or data architects. Additionally, developers have the power and flexibility of XQuery, a standard SQL-like query language represented in XML, to directly access the distributed data sources, which are represented as integrated logical views.

BEA Liquid Data for WebLogic builds on existing IT infrastructure and leverages XML standards and BEA WebLogic® Platform™—the industry-leading application infrastructure—for easy and secure deployment in existing IT environments. BEA Liquid Data for WebLogic can be deployed as a standalone solution or within the framework of a larger project.

Liquid Data Capabilities

BEA Liquid Data for WebLogic provides the following capabilities:

- Real Time Data Access
- Share Corporate Assets
- Increase Developer Productivity
- Standards-Based

Real Time Data Access

- **Universal data access**—Using XML translators and optimized XML queries, BEA Liquid Data for WebLogic can retrieve data from legacy and client-server applications, enterprise applications, relational databases, Web Services, Web sites, flat files and XML files, and other data sources, both inside and outside of corporate firewalls. BEA Liquid Data for WebLogic returns results in XML for easy consumption by Web applications.
- **Abstract Data Views**—A virtual abstraction layer aggregates distributed data sources as an integrated, logical view. For developers, these logical views of aggregated data sets—or data views—can be thought of as a single, virtual database. Data views allow easy access to shared business entities without needing to deal with the underlying complexity of varying data structures, semantics, and access methods.

Share Corporate Assets

- **Reusable Views**—Information about customers, orders, and inventory is needed again and again, by a variety of groups within the organization. BEA Liquid Data for WebLogic enables views of data to be presented as services that can be customized and reused. It makes it easy to modify views when new data sources need to be added.

- **Leverage domain resources**—Data views are created by a organization's data domain expert or data architect, who can quickly examine source schemas, and identify needed columns or elements. Data views are created and published once and shared many times by developers accessing data views as simple services, making the role of data access completely transparent to the developer.

Increase Developer Productivity

- **Expose and share Web Services**—Data views can be shared as Web services with other applications and departments by creating and sharing corporate entities (like customer or product profiles). The information is automatically tagged and organized for presentation by Web applications and can be invoked by a Web service or embedded rapidly and easily within an integrated application.
- **Dramatically reduce coding**—By providing developers with a high-level query language and visual development tools, BEA Liquid Data for WebLogic replaces months of custom coding with simple, familiar queries. Developers have the power and flexibility of a standard SQL-like language called XQuery for capabilities that can be written as familiar database queries, in only a few minutes.

Standards-Based

BEA Liquid Data for WebLogic takes full advantage of BEA WebLogic Server™, leveraging all the J2EE and XML standards, including new XML standards like XQuery. Adherence to these standards protects your investment.

Liquid Data Roles and Responsibilities

This section describes the roles, responsibilities, and resources for the following types of Liquid Data users:

- Data Architects
- Application Developers
- System Administrators

Each type of user has different tasks and tools for using the Liquid Data technology.

Data Architects

Data Architects know about the desired business entities to be created and the data sources that are required. Data Architects tend to be subject matter experts and have a deep understanding of the data, underlying schema, and relationships across the various data sources. They create the data views and stored queries used by the Application Developers, using either the Data View Builder tool or creating hand-coded XQuery queries. For more information, see *Building Queries and Data Views*.

Application Developers

Application Developers create applications that use the data views and stored queries, created by the Data Architects, to access real-time information. Application Developers can access data views and stored queries using the following mechanisms:

- Invoking queries in EJB client applications using the Liquid Data Query API.
- Invoking queries in JSP (Java Server Pages) clients using the Liquid Data JSP Tag Library
- Invoking queries that have been published as Web services

Application Developers can also write custom functions in Java code to extend Liquid Data functionality.

For more information, see *Invoking Queries Programmatically*.

System Administrators

System Administrators install, deploy, configure, and maintain the Liquid Data server. In addition to standard WebLogic Server administration tasks, an administrator uses the Liquid Data node in the Administration Console to perform the following kinds of tasks:

- Deploy Liquid Data in WebLogic domains.
- Configure access to data sources and queries.
- Set up and configure the Liquid Data server repository.
- Implement security using WebLogic Server's extensive compatibility security features, such as configuring users, groups, and access control lists (ACLs).
- Configure query caching to optimize performance.
- Monitor Liquid Data server operation, tune performance, and provide support.

For more information, see the Liquid Data *Administration Guide* and *Deploying Liquid Data*.

CHAPTER

2 Liquid Data Architecture

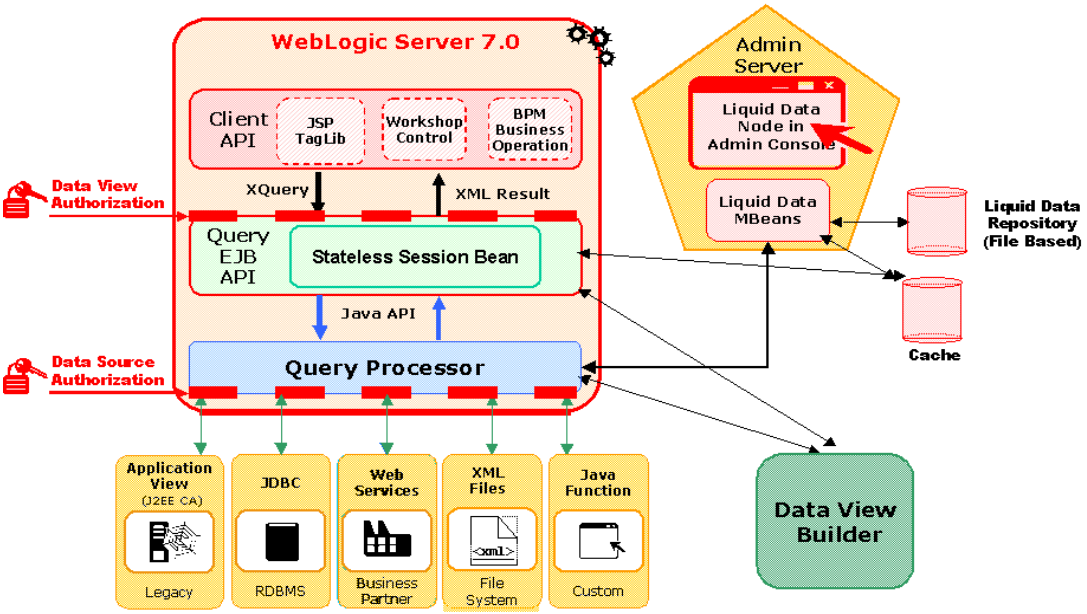
This topic provides an overview of the BEA Liquid Data for WebLogic™ architecture. It includes the following sections:

- Architecture Diagram
- Architecture Components

Architecture Diagram

The following diagram shows a component view of the Liquid Data architecture.

Figure 2-1 Liquid Data Architecture



Architecture Components

Liquid Data consists of the following core components:

- Server Components
- Client Components
- Developer Resources
- Integration with Other BEA Products (Optional)

Server Components

This section describes the main server components for Liquid Data:

- BEA WebLogic Server
- Liquid Data Server
- Liquid Data Server Repository

At a minimum, Liquid Data requires the Liquid Data Server, BEA WebLogic Server™, and the server repository.

BEA WebLogic Server

WebLogic Server is the platform upon which Liquid Data is built. WebLogic Server is the #1 application server on the market, providing the scalability, reliability, and security needed to run a global, networked environment—and all in one simple and extensible architecture. WebLogic Server incorporates comprehensive standards-based services for the development, deployment, and management of distributed Web-based applications. For comprehensive information about BEA WebLogic Server, see the WebLogic Server documentation at the following URL:

http://edocs.bea.com/wls/docs70/index.html

Liquid Data Server

The Liquid Data Server runs as an application in the WebLogic Server environment. The Liquid Data Server processes query requests from Liquid Data clients. It handles incoming query requests, translates the XQuery into an optimized algebraic plan, translates the algebraic plan into a physical plan, executes the physical plan against the data sources, and retrieves the query results.

Liquid Data Server Repository

The Liquid Data Server repository is the central location for stored queries, data views, source and target schemas, XML data, generated Web services, and custom function libraries. For more information, see "Managing the Liquid Data Repository" in the Liquid Data *Administration Guide*.

Client Components

This section describes the main client components for Liquid Data:

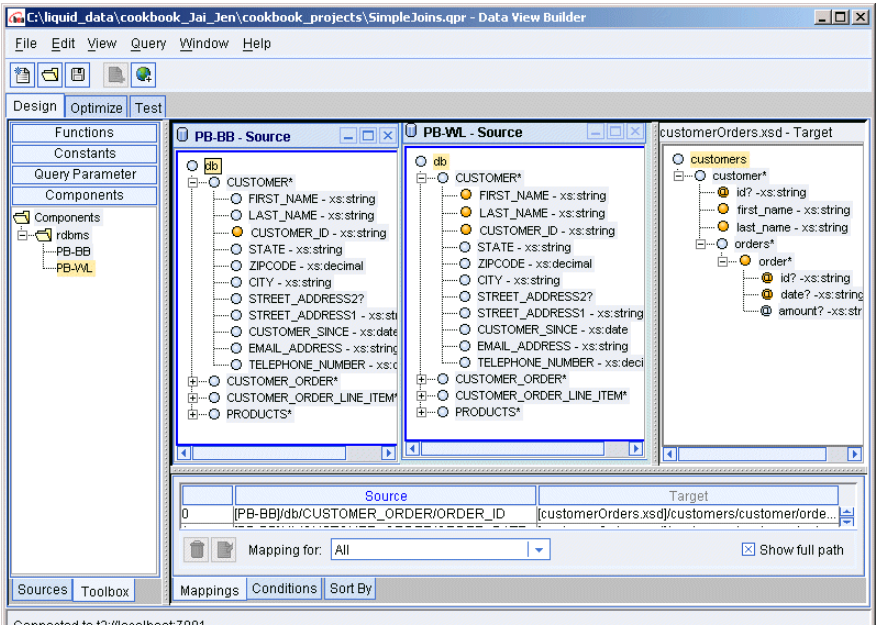
- Data View Builder
- Administration Console

In addition, other software can act as clients to the Liquid Data server by invoking queries with the Liquid Data Query API, such as custom EJB or JSP applications, Web services clients, BEA WebLogic Portal™, BEA WebLogic Workshop™, and the Business Process Management component of BEA WebLogic Integration™. For more information, see "Developer Resources" on page 2-6.

Data View Builder

The Data View Builder is a GUI-based tool for designing and generating XML-based queries (in XQuery syntax). Users can then run the queries—from within the Data View Builder, at a command line, or from within an application—against heterogeneous data sources to get logical views into the stored information. The Data View Builder provides a pictorial, drag-and-drop schema mapping approach to query design and construction.

Figure 2-2 Data View Builder

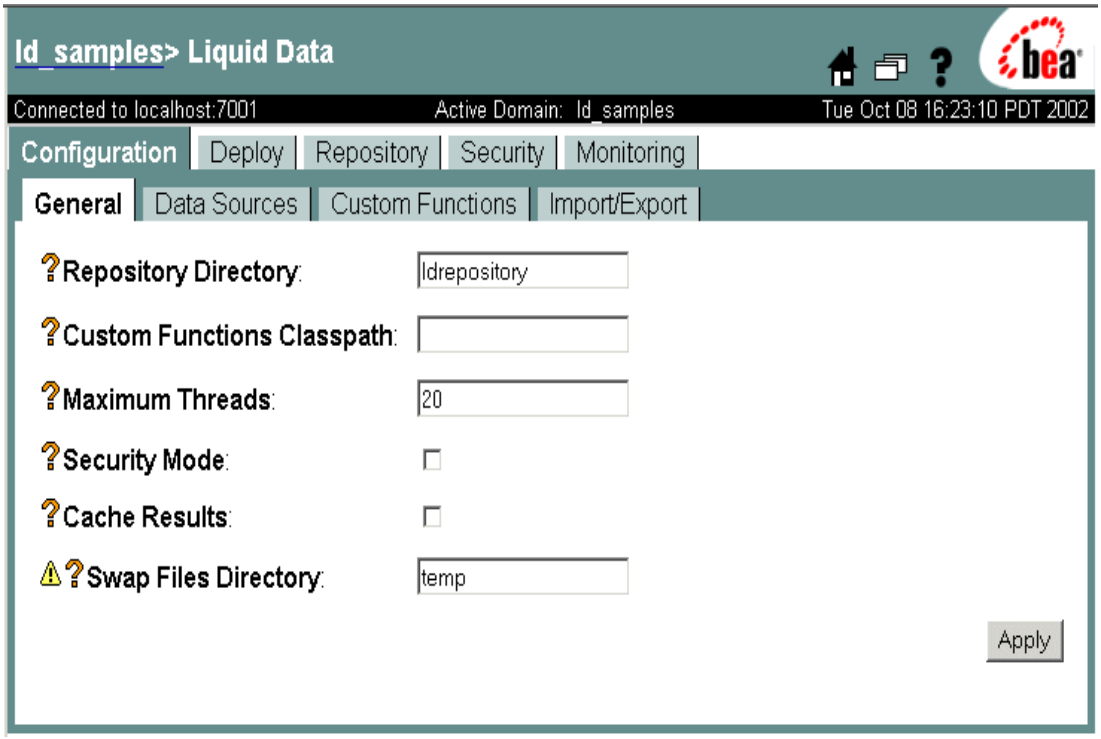


Using the Data View Builder frees Data Architects from needing to focus on the intricacies of query languages so that they can give full attention to information design, the conceptual synthesis of information coming from multiple sources, and the content and shape of the information they want in the query result or *target*. In this way, users can to directly access distributed, heterogeneous data sources as "integrated logical views." For more information about the Data View Builder, see *Building Queries and Data Views*.

Administration Console

The WebLogic Server Administration Console is a comprehensive, Web-based administration tool for configuring and monitoring WebLogic servers. Liquid Data extends the WebLogic Server Administration Console to provide additional tabs, accessible via a Liquid Data node, that are used to configure and manage Liquid Data servers deployed in the WebLogic Server environment.

Figure 2-3 Tabs in the Liquid Data Node in the Administration Console



System Administrators use the tabs on the Liquid Data node to configure access to data sources and custom functions, manage the server repository, configure query results caching, implement security, generate Web services, and other tasks. For more information, see the Liquid Data *Administration Guide*.

Developer Resources

This section describes the main developer resources available in Liquid Data:

- Liquid Data Query API
- Queries Published As Web Services
- Custom Functions

For complete information about Liquid Data developer resources, see *Invoking Queries Programmatically*.

Liquid Data Query API

Liquid Data provides an API (application programming interface) that allows client Java applications to submit queries to the Query Engine for processing. Liquid Data supports two types of clients:

- EJB clients can invoke queries on Liquid Data query EJBs, as described in "Invoking Queries in EJB Clients" in *Invoking Queries Programmatically.*
- JSP (Java Server Pages) clients can invoke queries using the Liquid Data JSP tag library, as described in "Invoking Queries in JSP Clients" in *Invoking Queries Programmatically.*

In addition to custom client applications, the Liquid Data Query API allows other BEA products—WebLogic Portal, WebLogic Workshop, WebLogic Web services, and the Business Process Management component of WebLogic Integration—to invoke queries on the Liquid Data server transparently.

Queries Published As Web Services

Liquid Data publishes queries as Web services that can be accessed by Web service clients. For more information, see the following documents:

- "Generating and Publishing Web Services" in the Liquid Data *Administration Guide*
- "Invoking Queries in Web Service Clients" in *Invoking Queries Programmatically*

Custom Functions

The Data View Builder provides a set of standard functions to use when creating data views and queries. Application Developers can also define *custom function*s to extend the power and functionality of Liquid Data. Queries can invoke custom functions during query execution just as they can standard functions. For more information, see "Creating Custom Functions" in *Invoking Queries Programmatically*.

Integration with Other BEA Products (Optional)

Liquid Data is an option that can be added to the following BEA products:

- BEA WebLogic Portal
- BEA WebLogic Workshop
- BEA WebLogic Integration

WebLogic Integration is not required except when queries need to access data sources via packaged JCA/WebLogic Integration-based adapters.

BEA WebLogic Portal

Liquid Data provides a cost effective and quick mechanism to access and aggregate information into a portal. It simplifies data aggregation and caches relevant data into portlets, offering a rapid and cost effective way for real-time data access. For more information, see "Invoking Queries in BEA WebLogic Portal Applications" in *Invoking Queries Programmatically* and "Deploying Liquid Data in a WebLogic Portal Domain" in "Deployment Tasks" in *Deploying* Liquid Data.

BEA WebLogic Workshop

WebLogic Workshop users can add generated Liquid Data Web services as service controls. For more information, see "Generating and Publishing Web Services" in the Liquid Data *Administration Guide* and "Deploying Liquid Data in a WebLogic Workshop Domain" in "Deployment Tasks" in *Deploying Liquid Data*.

BEA WebLogic Integration

Liquid Data extends the value of WebLogic Integration by seamlessly integrating disparate data views into enterprise business processes. Liquid Data can leverage the WebLogic Integration adapters to access even broader data sources, including packaged applications. These broader, more complete information views can then be easily re-used within the integration environment, to support information capture, visibility and decision-making within processes.

For comprehensive information about BEA WebLogic Integration, see the WebLogic Integration documentation at the following URL:

http://edocs.bea.com/wli/docs70/index.html

If queries need to access data sources via packaged JCA/WebLogic Integration-based adapters, then the Application View Console is required. The Application View Console is used to design and organize *application views*, which are business-level interfaces that provide access to functionality in enterprise applications. Application views are used in the context of the Application Integration component of WebLogic Integration. Liquid Data can use application views as data sources, allowing queries to retrieve information from enterprise applications. For more information, see "Defining an Application View" in *Using Application Integration* in the WebLogic Integration documentation.

For more information about Liquid Data and WebLogic Integration, see "Invoking Queries in Business Process Manager Applications" in *Invoking Queries Programmatically* and "Deploying Liquid Data in a WebLogic Integration Domain" in "Deployment Tasks" in *Deploying* Liquid Data.

CHAPTER

3 Liquid Data Concepts

This section provides an overview of core BEA Liquid Data for WebLogic™ concepts and also some usage information. It contains the following sections:

- Liquid Data Implements the XQuery Standard
- Definitions of Key Terms

For an overview of Liquid Data components, see Chapter 2, "Liquid Data Architecture."

Liquid Data Implements the XQuery Standard

This topic describes XQuery and how Liquid Data implements the XQuery standard. It contains the following sections:

- About XQuery
- XQuery Links
- XQuery Example

Liquid Data provides a scalable, highly available platform for building and processing XQuery queries. Data Architects can use the Data View Builder tool to create XQuery queries graphically (without needing to code queries in XML), as well as create ad hoc XQuery queries by hand. Client applications can use the Liquid Data Query API or generated Liquid Data Web services to invoke stored XQuery queries programmatically.

About XQuery

XQuery is a standard query language, published by the W3C (World Wide Web Consortium), that uses XML (Extensible Markup Language) notation to define query requests and handle query results. XQuery is designed to be an efficient language in which queries are concise and easily understood. XQuery is flexible enough to query a broad spectrum of data sources, including relational databases, XML documents, Web services, packaged applications, and legacy systems.

Applications of XQuery include filtering a document to produce a table of contents, providing joins across multiple data sources, grouping and aggregates, and queries based on sequential relationships in documents. Joins, in particular, represent a powerful and likely principal use of XQuery in the enterprise. The XQuery language is derived from various sources, including SQL. It even provides a For-Let-Where-Return (FLWR - pronounced "flower") expression. Developers familiar with SQL will find XQuery very easy to learn.

XQuery Links

For more information about XQuery, see the following topics on the W3C's Web site:

- XQuery 1.0: An XML Query Language at the following URL:
  `http://www.w3.org/TR/xquery/`
- XML Query Use Cases at the following URL:
  `http://www.w3.org/TR/xmlquery-use-cases`
- XML Path Language (XPath) 2.0 at the following URL:
  `http://www.w3.org/TR/xpath20/`
- XQuery 1.0 and XPath 2.0 Data Model at the following URL:
  `http://www.w3.org/TR/query-datamodel/`
- XQuery 1.0 and XPath 2.0 Functions and Operators at the following URL:
  `http://www.w3.org/TR/xquery-operators/`

XQuery Example

This section shows an example of XQuery code and sample query results from the Avitek sample. For more information about the Avitek sample, see *Running the Samples*.

Sample XQuery Query

The following code example shows the use of XQuery coding in a sample Order query.

Listing 3-1 XQuery for the Order Query from the Avitek Sample

```
{--     Generated by Data View Builder --}


<CustomerOrderReport>
{
   for $PB-WL.CUSTOMER_1 in document("PB-WL")/db/CUSTOMER
   where ($PB-WL.CUSTOMER_1/CUSTOMER_ID  eq  "CUSTOMER_1")
   return
   <customerOrder>
      <CUSTOMER>
         <FIRST_NAME>{ xf:data($PB-WL.CUSTOMER_1/FIRST_NAME) }</FIRST_NAME>
         <LAST_NAME>{ xf:data($PB-WL.CUSTOMER_1/LAST_NAME) }</LAST_NAME>
         <CUSTOMER_ID>{ xf:data($PB-WL.CUSTOMER_1/CUSTOMER_ID) }</CUSTOMER_ID>
         <STATE>{ xf:data($PB-WL.CUSTOMER_1/STATE) }</STATE>
         <EMAIL_ADDRESS>{ xf:data($PB-WL.CUSTOMER_1/EMAIL_ADDRESS) }</EMAIL_ADDRESS>
         <TELEPHONE_NUMBER>{ xf:data($PB-WL.CUSTOMER_1/TELEPHONE_NUMBER)
}</TELEPHONE_NUMBER>
      </CUSTOMER>
      <wireless_orders>
      {
         for $PB-WL.CUSTOMER_ORDER_2 in document("PB-WL")/db/CUSTOMER_ORDER
```

```
         where ($PB-WL.CUSTOMER_1/CUSTOMER_ID  eq  {--! ppright !--}
$PB-WL.CUSTOMER_ORDER_2/CUSTOMER_ID)

         return

         <CUSTOMER_ORDER>

            <ORDER_DATE>{ xf:data($PB-WL.CUSTOMER_ORDER_2/ORDER_DATE) }</ORDER_DATE>

            <ORDER_ID>{ xf:data($PB-WL.CUSTOMER_ORDER_2/ORDER_ID) }</ORDER_ID>

            <CUSTOMER_ID>{ xf:data($PB-WL.CUSTOMER_ORDER_2/CUSTOMER_ID)
}</CUSTOMER_ID>

            <SHIP_METHOD>{ xf:data($PB-WL.CUSTOMER_ORDER_2/SHIP_METHOD)
}</SHIP_METHOD>

            <TOTAL_ORDER_AMOUNT>{ xf:data($PB-WL.CUSTOMER_ORDER_2/TOTAL_ORDER_AMOUNT)
}</TOTAL_ORDER_AMOUNT>

         </CUSTOMER_ORDER>

      }

      </wireless_orders>

      <broadband_orders>

      {

         for $XM-BB-CO.CUSTOMER_ORDER_3 in document("XM-BB-CO")/db/CUSTOMER_ORDER

         where ($PB-WL.CUSTOMER_1/CUSTOMER_ID  eq  {--! ppright !--}
$XM-BB-CO.CUSTOMER_ORDER_3/CUSTOMER_ID)

         return

         <CUSTOMER_ORDER>

            <ORDER_DATE>{ xf:data($XM-BB-CO.CUSTOMER_ORDER_3/ORDER_DATE)
}</ORDER_DATE>

            <ORDER_ID>{ xf:data($XM-BB-CO.CUSTOMER_ORDER_3/ORDER_ID) }</ORDER_ID>

            <CUSTOMER_ID>{ xf:data($XM-BB-CO.CUSTOMER_ORDER_3/CUSTOMER_ID)
}</CUSTOMER_ID>

            <SHIP_METHOD>{ xf:data($XM-BB-CO.CUSTOMER_ORDER_3/SHIP_METHOD)
}</SHIP_METHOD>

            <TOTAL_ORDER_AMOUNT>{
xf:data($XM-BB-CO.CUSTOMER_ORDER_3/TOTAL_ORDER_AMOUNT) }</TOTAL_ORDER_AMOUNT>

         </CUSTOMER_ORDER>

      }

      </broadband_orders>

   </customerOrder>
```

```
}
</CustomerOrderReport>
```

Sample Query Results

The following listing shows the results returned in XML format from the sample Order query.

Listing 3-2 Results for the Order Query from the Avitek Sample

```
<CustomerOrderReport><customerOrder><CUSTOMER><FIRST_NAME>JOHN_1<
/FIRST_NAME><LAST_NAME>KAY_1</LAST_NAME><CUSTOMER_ID>CUSTOMER_1</
CUSTOMER_ID><STATE>TX</STATE><EMAIL_ADDRESS>abc@abc.com</EMAIL_AD
DRESS><TELEPHONE_NUMBER>4081231234</TELEPHONE_NUMBER></CUSTOMER><
wireless_orders><CUSTOMER_ORDER><ORDER_DATE>2002-03-06-08:00</ORD
ER_DATE><ORDER_ID>ORDER_ID_1_0</ORDER_ID><CUSTOMER_ID>CUSTOMER_1<
/CUSTOMER_ID><SHIP_METHOD>AIR</SHIP_METHOD><TOTAL_ORDER_AMOUNT>10
00</TOTAL_ORDER_AMOUNT></CUSTOMER_ORDER><CUSTOMER_ORDER><ORDER_DA
TE>2002-03-06-08:00</ORDER_DATE><ORDER_ID>ORDER_ID_1_1</ORDER_ID>
<CUSTOMER_ID>CUSTOMER_1</CUSTOMER_ID><SHIP_METHOD>AIR</SHIP_METHO
D><TOTAL_ORDER_AMOUNT>2000</TOTAL_ORDER_AMOUNT></CUSTOMER_ORDER><
/wireless_orders><broadband_orders><CUSTOMER_ORDER><ORDER_DATE>20
02-04-09</ORDER_DATE><ORDER_ID>ORDER_ID_1_0</ORDER_ID><CUSTOMER_I
D>CUSTOMER_1</CUSTOMER_ID><SHIP_METHOD>AIR</SHIP_METHOD><TOTAL_OR
DER_AMOUNT>1000.00</TOTAL_ORDER_AMOUNT></CUSTOMER_ORDER><CUSTOMER
_ORDER><ORDER_DATE>2002-04-09</ORDER_DATE><ORDER_ID>ORDER_ID_1_1<
/ORDER_ID><CUSTOMER_ID>CUSTOMER_1</CUSTOMER_ID><SHIP_METHOD>AIR</
SHIP_METHOD><TOTAL_ORDER_AMOUNT>1500.00</TOTAL_ORDER_AMOUNT></CUS
TOMER_ORDER></broadband_orders></customerOrder></CustomerOrderRep
ort>
```

Definitions of Key Terms

This topic introduces the following key Liquid Data concepts:

- Data Sources
- Queries, Results, and Schemas
- Data Views
- Caching
- Scoping and Query Hints
- Custom Functions

For a description of Liquid Data components, see "Architecture Components" on page 2-3.

Data Sources

In Liquid Data, a *data source* is a source of information that can be queried. Liquid Data supports querying the following types of data sources:

- Relational databases (RDBMSs) via JDBC
- XML files
- Web services
- Application views, which are business-level interfaces the data in packaged application such as Siebel, PeopleSoft, or SAP. Application views are used in conjunction with the Application Integration component of WebLogic Integration.
- data views, which are the dynamic results of queries stored along with the queries that produce them

You use the WebLogic Administration Console to configure access to data sources. For more information, see the following topics in the Liquid Data *Administration Guide*:

- "Configuring Access to Relational Databases"
- "Configuring Access to XML Files"
- "Configuring Access to Web Services"
- "Configuring Access to Application Views"
- "Configuring Access to Data Views"

You can use the Data View Builder to design and build queries that query the data sources you have configured, or you can hand-code the XQueries. Queries are built based on source and target schemas that describe the structure of the data queried (source) and returned (target). For more information about creating queries, see *Building Queries and Data Views*.

Queries, Results, and Schemas

A *query* is a request for information based on explicit criteria. In Liquid Data, queries are created and used in compliance with the XQuery language specification. With the Data View Builder, you can create queries using drag-and-drop in source and target schemas.

Users can specify two types of queries:

- *Stored queries* are XQuery queries that have been saved in the Liquid Data repository.
- *Ad hoc queries* are queries that have not been stored in the Liquid Data repository but rather are passed to the Liquid Data server on the fly. An ad-hoc query is any hand-coded query or Data View Builder-generated query that is not a stored query.

A *result* is the information that a query yields. In Liquid Data, the query result is generated in XML format. The *target schema* is an XML schema or DTD that describes the shape (structure and legal elements) of the output data—that is, the result of a query. The Liquid Data server runs queries against source data and returns the result of a query the form in which you define for *target data*.

For more information about queries, results, and target schemas, see "Overview and Key Concepts" in *Building Queries and Data Views*.

Data Views

A *data view* represents the result of a query that provides a filtered view on the data and information being queried. As such, the data view (query result) is dynamic—it will continue to reshape itself based on any changes that occur in the data it is querying. Data views can be re-used as data sources by saving them in the Liquid Data server repository. For more information about data views, see "Overview and Key Concepts" and "Using Data Views as Data Sources" in *Building Queries and Data Views*.

Caching

Liquid Data caches information about commonly executed queries for subsequent, efficient retrieval, thereby enhancing overall system performance. Liquid Data supports two types of caching:

- The *query plan cache* is a memory-based cache that stores the query plans of commonly executed queries.
- The *result set cache* is a database-based cache that stores the results of commonly executed queries.

For more information, see "Configuring the Query Results Cache" in the Liquid Data *Administration Guide*.

Scoping and Query Hints

The following techniques help Liquid Data expedite the processing of queries more quickly and efficiently:

- *Scoping* helps clarify which part of a data view is the focal point for a particular condition in a query, affecting the placement of a "where" clause in the XQuery generation. In most cases, scope is implicit and the query generator can determine what the desired result should be, while in some cases you might need to communicate your objectives explicitly. For more information, see "Understanding Scope in Basic and Advanced Views" in "Designing Queries" in *Building Queries and Data Views*.

- For join operations, a *query hint* specifies which join algorithm to use when running a query on the Liquid Data server. For more information, see "Optimizing Queries" in *Building Queries and Data Views*.

Custom Functions

Custom functions are user-defined functions that performed specialized tasks. The Data View Builder provides a set of standard functions for use in creating data views and queries. In addition, users can create custom functions, which are implemented in Java code, declared in a custom functions library definition (CFLD) file (in XML format), and then configured in the Administration Console. Once configured, custom functions show up as functions available for use in the Data View Builder. For more information about custom functions, see "Using Custom Functions" in *Invoking Liquid Data Queries Programmatically* and "Configuring Access to Custom Functions" in the Liquid Data *Administration Guide*.

CHAPTER

4 Liquid Data Quick Start

The following sections guide you through all the tasks required to design and implement a real-time data access and aggregation solution using BEA Liquid Data for WebLogic™. This topic is meant to provide a high-level view of the process. To read more detail about any particular task, follow the links provided to the document where that topic is discussed in full detail.

- Step 1. Install the Software
- Step 2: Explore the Liquid Data Samples
- Step 3. Define the Data Access and Aggregation Requirements and Scope
- Step 4. Configure Access to Data Sources
- Step 5. Create XQuery Queries
- Step 6: Invoke Queries Programmatically (Optional)
- Step 7. Create the Presentation Layer
- Step 8. Deploy the Data Access and Aggregation Solution

For a walkthrough of steps 3, 4, and 5, see Getting Started With Liquid Data.

Step 1. Install the Software

To use Liquid Data, you begin by installing the WebLogic Platform and the Liquid Data software according to the instructions in *Installing Liquid Data*.

Step 2: Explore the Liquid Data Samples

After you install the software, you can run the one-time samples configuration, launch the samples server, and then explore Liquid Data using the Liquid Data samples. For more information, see *Running the Samples*.

Step 3. Define the Data Access and Aggregation Requirements and Scope

Before you can create queries, you need to clarify the requirements and scope of your data access and aggregation solution, answering such questions as:

- Who is the audience for data access and aggregation?
- What kinds of problem statements/queries, in plain English, are you trying to solve?
- What are the data sources and their types?
- What is the required information *output*, also known as the *query result*? What specific content is required and how should it be presented (layout, order, and so on)? How should the output or query result be structured in the *target schema*? What source-to-target mappings do you need to create in order to shape the result as needed?

- What is the required information *input*? Which data sources can provide such information? What *types* of data sources are involved—relational databases, XML files, Web services, application views, or data views? What is required to obtain access to these data sources? What is the structure of the data sources (*source schemas*)? What *conditions* do you need to define in a query in order to obtain the appropriate view on the information?

Step 4. Configure Access to Data Sources

Before you can create queries, you must configure access to the data sources that provide these queries with the required content. You use the Liquid Data node in the Administration Console to configure access to data sources, via *data source descriptions*, as described in the following topics in the Liquid Data *Administration Guide*:

- "Configuring Access to Relational Databases"
- "Configuring Access to XML Files"
- "Configuring Access to Web Services"
- "Configuring Access to Application Views"
- "Configuring Access to Data Views"

Certain data source types, such as relational databases and application views, require additional configuration tasks that are also described in their respective sections. You might also need to perform other configuration tasks described elsewhere in the Liquid Data *Administration Guide*.

Step 5. Create XQuery Queries

Once you have configured access to data sources, you can create XQuery queries to retrieve the information needed for your data access and aggregation solution. You can either use the Data View Builder to generate the XQuery for you (based on a design

you construct using its graphical tools), or you can hand code the XQuery. Either way, you use the Data View Builder to save queries as stored queries in the Liquid Data server repository. For more information, see *Building Queries and Data Views*.

Step 6: Invoke Queries Programmatically (Optional)

Client applications can invoke Liquid Data queries in the following ways:

- The Liquid Data Query API (Application Programming Interface) allows EJB and JSP client applications to invoke queries programmatically. Developers can create custom Java applications or build JSP pages. In addition, other BEA software—WebLogic Portal, the Business Process Management component of WebLogic Integration, WebLogic Web Services, and WebLogic Workshop—can invoke queries on the Liquid Data Server, providing real-time integration with Liquid Data.
- Web service clients can invoke Liquid Data queries using Web services that have been generated in the Administration Console from stored queries.

For more information, see "*Invoking Queries Programmatically*."

Step 7. Create the Presentation Layer

In Liquid Data the *presentation layer* is the software component that handles the layout, formatting, and display of query results to users. Liquid Data provides a JSP (Java Server Pages) tag library that you can use to create the presentation layer in JSP pages. The Liquid Data Tag Library can be used with both stored queries and ad hoc queries. For more information, see *Creating the Presentation Layer*.

Step 8. Deploy the Data Access and Aggregation Solution

After you have created all the components of your data access and aggregation solution, you need to deploy it in a production environment. In addition to defining deployment requirements and designing the deployment, you need to deploy the Liquid Data Server, the server repository, and other resources in a WebLogic domain in a production environment. For more information, see "*Deploying Liquid Data*."

Index

P

Q

R

S

T

W

X