BEA Liquid Data for WebLogic™

Deployment Guide

Version 8.1
Document Date: July 2003
Revised: July 2004

Copyright

Restricted Rights Legend

Trademarks or Service Marks

Contents

About This Document

1. Introduction

2. Planning a Liquid Data Deployment

3. Deployment Tasks

4. Troubleshooting a Deployment

5. Tuning Liquid Data Performance

Index

About This Document

This document describes the overall process and the types of tasks necessary to deploy a BEA Liquid Data for WebLogic data integration solution in a production environment.

This document covers the following topics:

- Chapter 1, "Introduction," identifies resources associated with Liquid Data deployments and provides an overview of the deployment process.
- Chapter 2, "Planning a Liquid Data Deployment," defines deployment requirements and provides an overview of things to consider when designing Liquid Data deployments.
- Chapter 3, "Deployment Tasks," provides detailed instructions for the tasks involved in deploying Liquid Data in a production environment, including: defining deployment requirements, designing a deployment, and deploying Liquid Data in various configurations (standalone, multi-node, and clustered deployments) in different types of domains (BEA WebLogic Platform, BEA WebLogic Integration, BEA WebLogic Server, BEA WebLogic Workshop, and BEA WebLogic Portal).
- Chapter 4, "Troubleshooting a Deployment," identifies resources to assist with troubleshooting and provides instructions on how to troubleshoot out of memory problems
- Chapter 5, "Tuning Liquid Data Performance," describes Liquid Data performance factors, identifies ways to monitor Liquid Data performance, and provides instructions for tuning performance in a Liquid Data deployment.

What You Need to Know

This document is intended mainly for administrators who are responsible for deploying and maintaining Liquid Data in a production environment. Administrators must know how to navigate the WebLogic Server Administration Console, be able to complete the installation tasks in the Liquid Data *Installation Guide* and the administrative tasks in the *Administration Guide* and have BEA platform knowledge.

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 BEA Liquid Data for WebLogic 8.1.

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.

Convention	Item

`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
{ }	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.

Convention	Item

. . .	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

Introduction

Deployment is the process of setting up BEA Liquid Data for WebLogic in a production environment. This topic introduces Liquid Data deployment. It contains the following sections:

- Liquid Data Deployment Considerations
- WebLogic Platform Configuration Wizard
- Resources to Deploy
- Resources to Configure

Liquid Data Deployment Considerations

While Liquid Data can be adequately evaluated on a single-server system, a multiple server configuration is generally considered appropriate for a production environment.

In Chapter 2, "Planning a Liquid Data Deployment", you will find suggested Liquid Data deployment designs with cross-references to WebLogic Server deployment documentation.

When Liquid Data is used for significant numbers of queries or significantly large and resource-intensive queries, you should deploy Liquid Data into its own domain or domains. This will provide the best opportunities to analyze and tune the system. Similarly, when your needs grow, it will be easier to identify those needs if Liquid Data is established with its own set of servers.

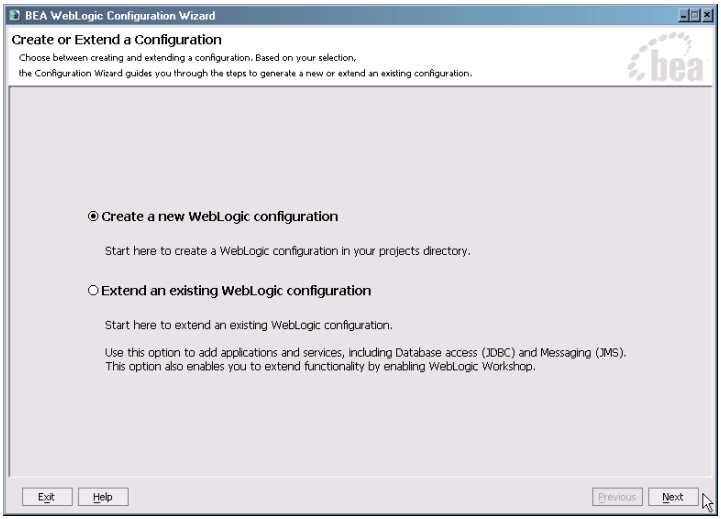
On the other hand, when Liquid Data is playing a supportive role for another application (such as WebLogic Integration or WebLogic Workshop) it can be deployed either in a new or an existing WebLogic 8.1 domain.

Initially, you may want to install Liquid Data as a single server system or a two-node cluster (if rollover is a high priority). In this way it will not be difficult to tell if a simple configuration is adequate or if additional nodes are needed.

WebLogic Platform Configuration Wizard

You should use the WebLogic Platform 8.1 Configuration Wizard to create and configure domains with or without Liquid Data.

Figure 1-1 Domain Configuration Wizard

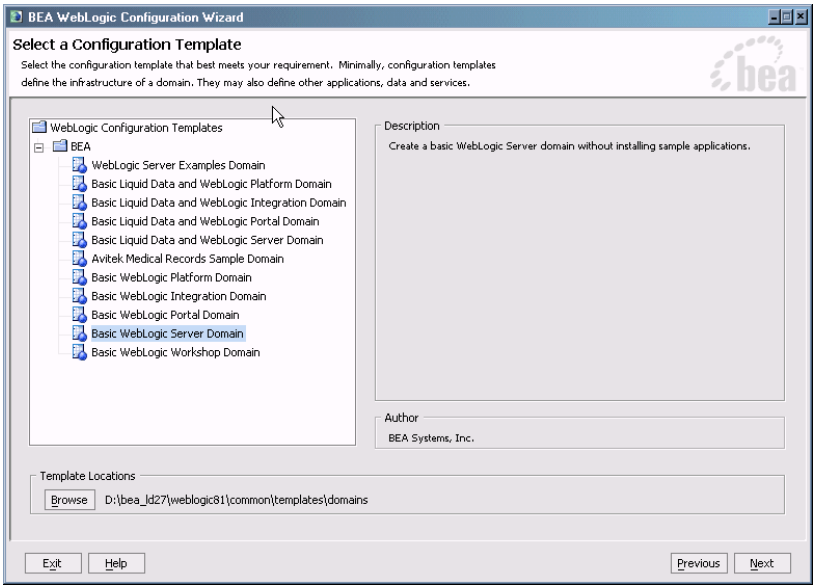


Creating New Liquid Data-enabled WebLogic Domains

After you install Liquid Data, the following deployment templates for creating a new domain become available from the Configuration Wizard:

- Basic Liquid Data and WebLogic Platform Domain
- Basic Liquid Data and WebLogic Integration Domain
- Basic Liquid Data and WebLogic Portal Domain
- Basic Liquid Data and WebLogic Server Domain

Figure 1-2 Liquid Data Templates Available in the Configuration Wizard



The Configuration Wizard provides both an Express and Custom configuration options. If you choose Custom, you can tailor your deployment for such items as:

- distribution across servers, clusters, and physical machines
- jdbc components such as connection pools, multipools, and data sources
- jms components such as stores, topics and queues
- modifying or specifying application, JDBC, and JMS component deployment servers and clusters

Extending an Existing WebLogic Domain for Liquid Data

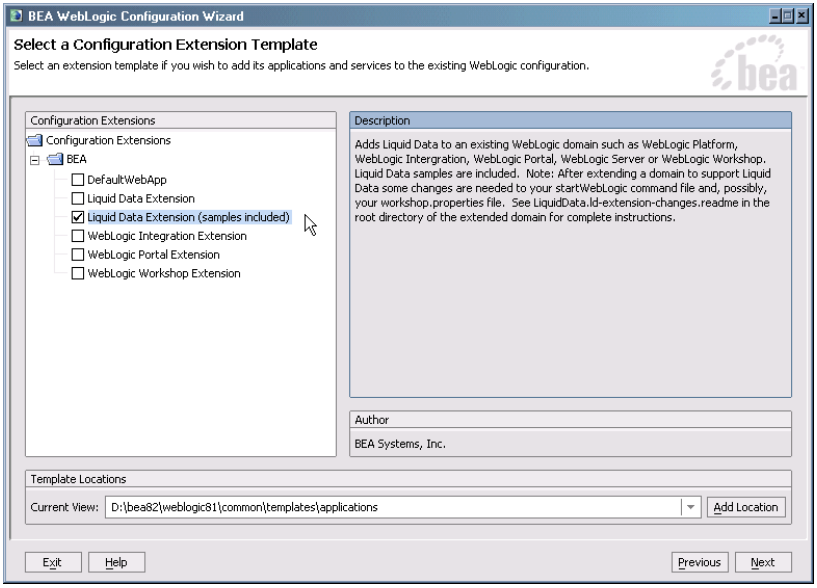
You can also extend any existing WebLogic Platform 8.1 domain to support Liquid Data.

For example, to create a Liquid Data + WebLogic Workshop domain you would use the Configuration Wizard to:

1. Create and configure a WebLogic Workshop domain (if you do not already have on).
2. Restart the Configuration Wizard, selecting the Extend an Existing WebLogic Configuration option

3. Check the Liquid Data Extension (samples included) checkbox.
4. Click Configure to extend the WebLogic Workshop domain to include Liquid Data.

Figure 1-3 Dialog Box for Extending a WebLogic Domain for Liquid Data



If the “samples included“ option is chosen, the Liquid Data samples are installed to the selected domain.

Note: When extending a domain using the Configuration Wizard, it is highly recommended that you extend a domain one product at a time. If additional extension are needed, rerun the Configuration Wizard.

Minor script editing is needed to complete a deployment.

Overview of Deployment Steps

The process of deploying Liquid Data involves the following basic steps:

1. Make sure that WebLogic Platform and Liquid Data are installed.
2. Determine whether to deploy to a new or an existing domain.
3. If you are deploying Liquid Data to a new domain review "Defining Deployment Requirements" on page 2-1 and "Designing Deployments" on page 2-3. Such a review can help ensure that your deployment architecture best suits your long-term business requirements.
4. Install WebLogic Server and Liquid Data according to the instructions in Installing Liquid Data.
5. Using the WebLogic Platform Configuration Wizard create or extend the appropriate WebLogic Platform domains with Liquid Data, as described in Chapter 3, "Deployment Tasks."
6. Properly set up security for your newly created or extended domain. See "Security in Liquid Data" in the *Administration Guide* for detailed information and further references.
7. Start the servers in the deployment domain(s) according to the instructions in "Starting and Stopping the Server" in the Liquid Data *Administration Guide*.
8. Start the WebLogic Administration Console and configure the resources described in "Resources to Configure" on page 1-7 as applicable to your deployment.
9. Monitor the ongoing operation of the Liquid Data server according to the instructions in "Monitoring the Server" in the Liquid Data *Administration Guide*.
10. Tune performance on the Liquid Data server, as necessary, according to the instructions in Chapter 5, "Tuning Liquid Data Performance."

These steps provide only a summary of the steps required to deploy Liquid Data in a production environment.

Note: If your domain will include applications that use the Liquid Data Control, you need to update the ldcontrol.properties file. For details, see "Moving Your Liquid Data Control Applications to Production" in the *Application Developer's Guide*.

`http://edocs.bea.com/liquiddata/docs81/program/ld_control.html#deployControl`

Deployment References in WebLogic Platform Documentation

In addition to this document, BEA WebLogic Platform documentation provides a comprehensive deployment page at the following URL: http://e-docs.bea.com/platform/docs81/confgwiz/index.html

Resources to Deploy

The following table summarizes the primary resources involved in a Liquid Data deployment:

Table 1-4 Resources to Deploy in a Liquid Data Deployment

Resource	Description
WebLogic Platform	The complete WebLogic Platform includes the following components: • WebLogic Server • WebLogic Workshop • WebLogic Integration • WebLogic Portal **Note:** At a minimum, the WebLogic Server must be deployed. Other components are deployed as needed.
Liquid Data software	Liquid Data is deployed in an enterprise archive (`LDS.ear`) file format on a single WebLogic Server instance or, in a clustered deployment, across all servers in the cluster.
Data sources	Data sources used in Liquid Data queries must be accessible to the Liquid Data server.

Resources to Configure

Through the Configuration Wizard, you can configure the following resources as applicable for your deployment:

Table 1-5 Liquid Data Resources to Configure

Resource	Description
Liquid Data server settings and server repository	You need to configure server settings on the General tab in the Liquid Data node of the WebLogic Administration Console, especially the Liquid Data repository location. If you are creating a production implementation from a test deployment, you will probably want to use the Liquid Data Import/Export options to recreate your data source descriptions to new server locations. For instructions, see "Configuring Liquid Data Server Settings" and "Managing the Liquid Data Server Repository" in the Liquid Data *Administration Guide*.
Data source descriptions	For each data source accessed in a Liquid Data query, you need to add a data source description using the Data Sources tab in the Liquid Data node of the WebLogic Administration Console. Additional configuration tasks are required for certain data source types. For instructions, 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" • Configuring Access to Custom Functions • Configuring Access to Delimited Files

Table 1-5 Liquid Data Resources to Configure (Continued)

Resource	Description
Results caching	If you want to cache results for stored queries in this deployment, you must explicitly enable results caching on the General tab in the Liquid Data node of the WebLogic Administration Console. In addition, for each stored query that you want cached, you need to explicitly configure the caching policy. For instructions, see "Configuring the Query Results Cache" in the Liquid Data *Administration Guide*.
Custom functions	If custom functions are used in this deployment, you need to create a custom function description for each custom function and add certain files to the server repository. For instructions, see "Configuring Access to Custom Functions" in the Liquid Data *Administration Guide*.

Note: You can copy a Liquid Data repository and configuration information from another, fully configured Liquid Data server according to the instructions in "Copying a Liquid Data Configuration From Server to Server" on page 3-12.

CHAPTER 2

Planning a Liquid Data Deployment

This topic describes the tasks involved in deploying BEA Liquid Data for WebLogic in a production environment. It includes the following sections:

- Defining Deployment Requirements
- Designing Deployments

Defining Deployment Requirements

When you are deploying Liquid Data to a new domain understanding your deployment requirements and needs are a critical first step in the process of deploying Liquid Data in a production environment.

When planning a deployment, consider the following issues:

- **Business requirements.** Identify and prioritize business requirements in a production environment, such as:
  - system performance required at anticipated peak loads
  - high availability and failover requirements
  - security and data access requirements
- **Resources to be deployed.** Identify the components of the BEA WebLogic Platform to use.

  At a minimum, BEA WebLogic Server and Liquid Data need to be deployed on at least one server.
- **Production environment.** Identify the hardware and software components of the production environment, including the network infrastructure, server nodes, and data sources.

- **Usage requirements.** Characterize how users are likely to utilize the system.

  Which types of queries (stored or ad hoc queries) will they run? How frequently will users be running different types of queries? Which queries will be run more frequently or less frequently than others? What are the anticipated peak loads for processing concurrent query requests?

- **Data source requirements.** Determine how to access any data sources used in Liquid Data queries.

- **Security requirements.** Determine which users need access to Liquid Data resources and the level of access they need. See "Security in Liquid Data" in the *Administration Guide* for detailed information and further references.

- **Licensing requirements.** Determine the BEA software licenses needed to implement your deployment solution.

Clarifying the specific requirements for your deployment provides the knowledge you will need to design your deployment.

Deployable Liquid Data Components

Liquid Data components are contained in the `LDS.ear` file.

These components can be deployed from the Liquid Data node of the WebLogic Administration Console. Table 2-1 lists the deployable Liquid Data components.

Table 2-1 Contents of the LDS.ear file

Component	Description	Deployment Target
`ejb_query.jar`	EJB that contains all query-related classes, including the Query Processor and stateless session bean. For more information, see the Liquid Data Javadoc.	For a multi-node and clustered deployment, deploy to each Managed Server.
`ejb_qbc.jar`	EJB for Data View Builder to obtain configuration information from the Liquid Data server.	For a multi-node and clustered deployment, deploy to each Managed Server.
`XMediator.war`	Initializes the Liquid Data server.	For a multi-node and clustered deployment, deploy to each Managed Server.
`ldconsole.war`	Liquid Data configuration tabs that appear in the WebLogic Server Administration Console.	Always deploy to an Administration Server.

Table 2-1 Contents of the LDS.ear file

Component	Description	Deployment Target
`cacheEjb.jar`	EJB that manages results caching for stored queries that are configured for results caching. For more information, see "Configuring the Query Results Cache" in the *Administration Guide*.	For a multi-node and clustered deployment, deploy to each Managed Server.
`ldcacheListener.war`	Listener application that listens to notifications from the MBean for changes to the global cache status (enabled or disabled), changes to a stored query that is cached, and changes to the cache policy for a stored query.	For a multi-node and clustered deployment, deploy to each Managed Server.

For details on using the Administration Console to deploy Liquid Data components see "Deploying Liquid Data Components" in the *Administration Guide*.

Designing Deployments

This topic describes how to design a Liquid Data deployment for the following types of domains:

- Single Domain Deployments
  - Single Server Deployments
  - Multi-Node Deployments
  - Clustered Deployments
  - Multi-Domain Deployments

The deployment architecture that you use depends on the requirements for your particular production environment, as described in "Defining Deployment Requirements" on page 2-1.

Single Domain Deployments

The following sections describe single domain deployments:

- Single Server Deployments
- Multi-Node Deployments

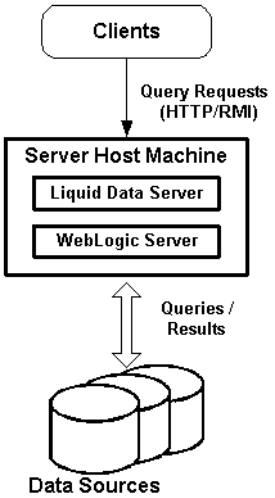
- Clustered Deployments

Single Server Deployments

In a single server (standalone) deployment, the Liquid Data software is installed on a standalone WebLogic Server. This design is the simplest to set up and it is the one suggested for use in a development environment or in a production environment in which Liquid Data is the primary application and failover protection is not the top priority.

The following illustration shows Liquid Data and WebLogic Server deployed on a single server:

Figure 2-2 Liquid Data and WebLogic Platform Deployed on a Single Server



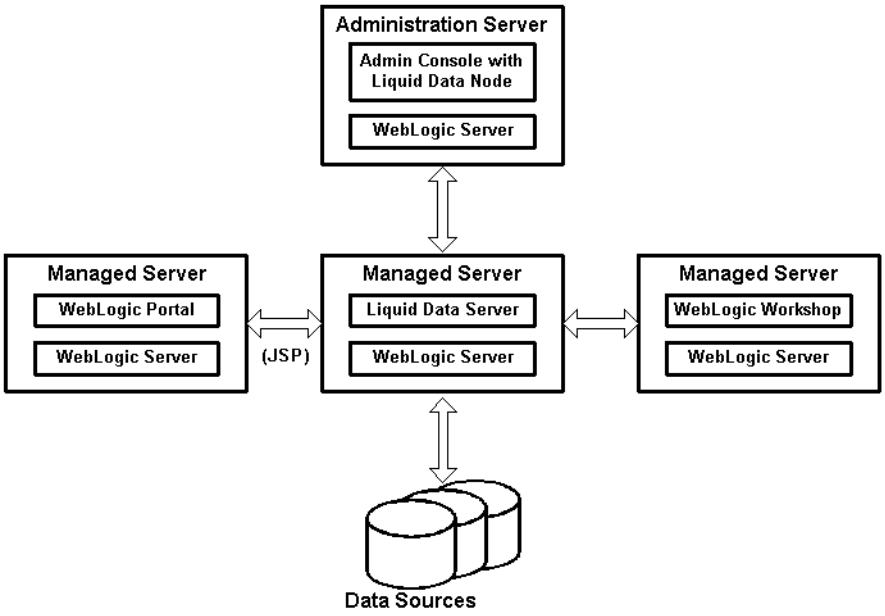
Multi-Node Deployments

A *multi-node deployment* distributes Liquid Data and WebLogic Platform software components across multiple server machines. A multi-node deployment allows you to run the various software components on dedicated servers, distributing resource contention across machines and optimizing system performance.

In each multi-node WebLogic Server domain, one WebLogic Server instance acts as the *Administration Server*—the server instance that configures, manages, and monitors all other server instances and resources in the domain. In a multi-node deployment, the Administration Server has administrative control over the domain and hosts the server repository. All other servers in the domain are configured as *Managed Servers*—servers that are managed by the Administration Server.

The following illustration shows Liquid Data deployed in a sample multi-node domain with three dedicated servers, one each for Liquid Data, WebLogic Portal, and WebLogic Workshop:

Figure 2-3 Liquid Data Deployed in a Multi-Node Domain



Clustered Deployments

A *clustered deployment* distributes Liquid Data and WebLogic Platform software components across multiple server machines. A *cluster* is a group of servers that work together to provide an application platform that is more powerful and reliable than a single server. A cluster appears to its clients as a single server but it is, in fact, a group of servers acting as one. In contrast to a multi-node deployment (in which Liquid Data and WebLogic Platform software components are installed separately on *dedicated* machines), in a clustered deployment, all Liquid Data and WebLogic Platform software components are installed on *every* machine.

A Liquid Data cluster is a deployment in which multiple copies of Liquid Data, referred to as *instances*, run simultaneously and work together to provide increased scalability and reliability. A Liquid Data cluster appears to clients to be a single Liquid Data instance. The server instances that constitute a cluster can run on the same machine, or can be located on different machines. You can increase a cluster's capacity by adding additional server instances to the cluster on an existing machine, or you can add machines to the cluster to host the incremental server instances. Each server

instance in a cluster must run the same version of WebLogic Server. For additional information about clustering, see "Using WebLogic Server Clusters" in the WebLogic Server documentation.

In each WebLogic Server domain, one WebLogic Server instance acts as the Administration Server—the server instance that configures, manages, and monitors all other server instances and resources in the domain. In a clustered deployment, the Administration Server has administrative control over the domain and hosts the server repository. All other servers in the domain are configured as *Managed Servers*—servers that are managed by the Administration Server. In a single instance deployment, the sole WebLogic Server instance is the Administration Server by default.

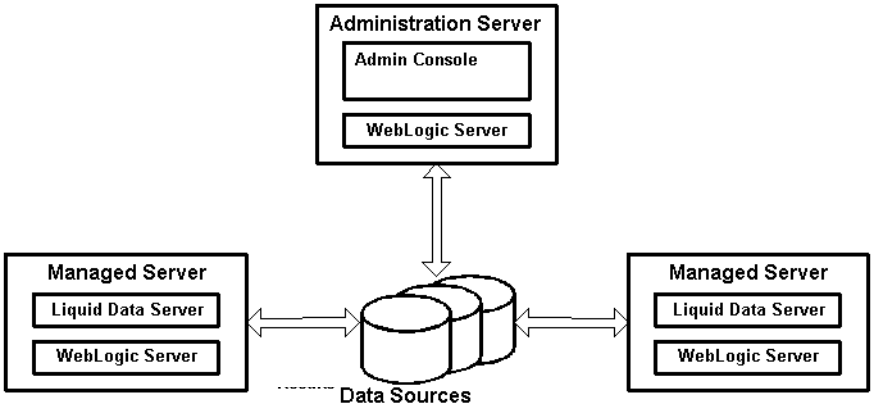
Note: In general, avoid configuring the Administration Server as part of the cluster.

In a clustered deployment, you select the algorithm to use for load balancing (round robin, weight-based, or random). Once configured, the load balancing mechanism directs incoming XQuery query requests to available Liquid Data server instances. For more information, see "Load Balancing in a Cluster" in *Using* WebLogic Server *Clusters* in the WebLogic Server documentation.

In a clustered Liquid Data deployment, if a server fails, the WebLogic cluster can redirect request processing to another Liquid Data server instance in the cluster. For more information, see "Replication and Failover for EJBs and RMIs" in "Failover and Replication in a Cluster" in *Using* WebLogic Server *Clusters* in the WebLogic Server documentation.

The following illustration shows Liquid Data deployed in a clustered domain.

Figure 2-4 Liquid Data Deployed in a Clustered Domain



This design is suggested for a deployment environment in which:

- The high volume of concurrent query requests necessitates additional machines in order to achieve system performance requirements.

- High availability through failover protection in the event of a server failure is a deployment requirement.

Before proceeding to deploy Liquid Data in a cluster, you should determine whether running Liquid Data on a dedicated server machine, as configured in "Multi-Node Deployments" on page 2-4, meets the performance and availability requirements for your deployment. Using the single server as a baseline, you can monitor run-time performance, conduct capacity planning, and test to determine whether a single machine provides sufficient performance at high query request loads.

Multi-Domain Deployments

In a multi-domain deployment, Liquid Data and the WebLogic Platform component are installed on separate WebLogic domains.

CHAPTER 3

Deployment Tasks

You can easily deploy Liquid Data into new or existing domain using the WebLogic Configuration Wizard. On Windows you can access the WebLogic Configuration Wizard from the Start menu:

```
Start —> BEA WebLogic Platform 8.1 —> Configuration Wizard
```

Alternatively, you can access the WebLogic Configuration Wizard directly.

Windows:

```
<wl_home>\common\bin\config.cmd
```

UNIX:

```
<wl_home>/common/bin/config.sh
```

When deploying a Liquid Data-enabled domain through the WebLogic Configuration Wizard, deployment of Liquid Data components is automated. This includes distributing contents of the `LDS.ear` file to the appropriate server. (In a multi-module deployment the `ldconsole.jar` should only be deployed on the Administration Server; all other modules should only deployed on Managed Servers. See "Deployable Liquid Data Components" on page 2-2 for details.)

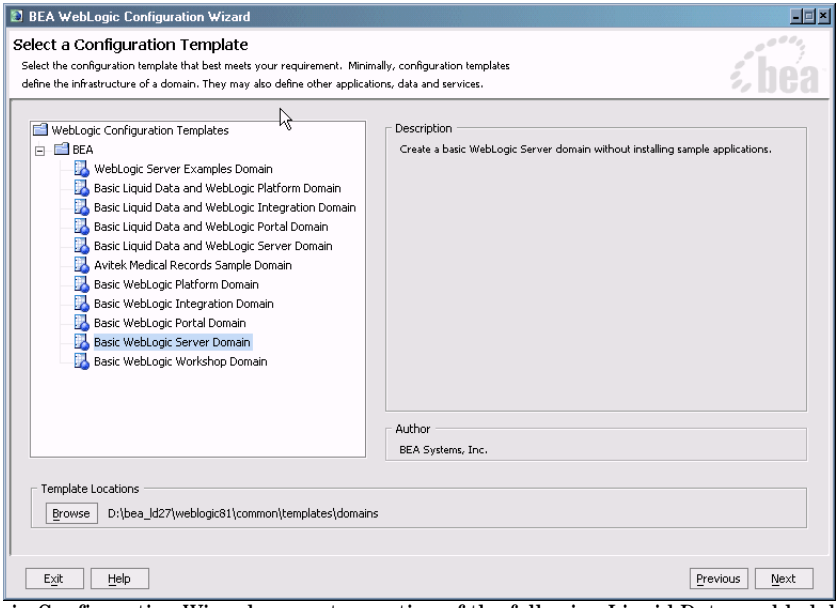
When you start the WebLogic Configuration Wizard, you are first asked whether you want to create a new WebLogic platform domain or extend an existing domain. If you choose to import into an existing domain, you will be able to navigate to the domain you wish to extend and set deployment configurations appropriately.

This chapter describes tasks involved in deploying BEA Liquid Data for WebLogic. In addition to deploying Liquid Data to new and existing domains, some special cases are described in detail. The following sections are included:

Deploying Liquid Data with a New WebLogic Domain

When Liquid Data is installed, it adds several configuration templates to the WebLogic Configuration Wizard.

Figure 3-1 Liquid Data Options in WebLogic Platform Configuration Wizard



The Domain Configuration Wizard supports creation of the following Liquid Data-enabled domains:

- Basic Liquid Data and WebLogic Platform Domain
- Basic Liquid Data and WebLogic Integration Domain
- Basic Liquid Data and WebLogic Portal Domain

- Basic Liquid Data and WebLogic Server Domain

When creating a new, Liquid Data-enabled domain, you have a choice between Express and Custom configuration. As the name implies, Express auto-selects most options to allow you to create a Liquid Data-enabled domain very quickly.

If you choose the Custom option, the WebLogic Configuration Wizard guides you through a series of dialog boxes that allow you to set or select:

- Managed servers, clusters, and machines options
- Database (JDBC) options
- Messaging options (such as stores, topics and queues)
- User and group permissions
- Windows Start menu options and entries
- Administration options
- Development or production mode
- a Java SDK

After using the WebLogic Configuration Wizard, follow these steps to complete the deployment process:

1. If you chose production mode when creating the domain, follow the steps described in "Modifying the Startup Script To Set the Administrative User Name" on page 3-11.
2. Start or restart WebLogic Server, logging in as the default user created at the time when the domain was generated.
3. If you are setting up a clustered deployment you need to decide how your managed servers will access a shared `ldrepository` directory.
   - If the deployed Liquid Data repository seldom if ever changes, you can simply set the `ldrepository` directory structure in the Liquid Data node of the WebLogic Administration Console (Configuration —> General —> Repository Directory) to a specific location and then copy the `ldrepository` directory and its contents to the same exact location on each managed server.

     For example of you had a cluster with two managed servers your `ldrepository` directory might be located at:

     Managed Server No. 1:

- `/opt/prod/bea_81/user_domains/ld_domain/ldrepository`

Managed Server No. 2:

- `/opt/prod/bea_81/user_domains/ld_domain/ldrepository`

The advantage of this approach is that you will get better performance with a local repository on each machine.

– Alternatively, you can change the `ldrepository` directory path so that it points to a shared volume to which all Managed Servers have access. On Windows, for example, you would point to a shared network drive mapped to a specific drive letter (such as `R:\ldrepos`).

In both cases you set the Liquid Data repository path using the Liquid Data node of the WebLogic Administration Console as follows:

a. Start or restart the Administration Console, logging in as appropriate.

b. In the left pane, click on the Liquid Data node, then click the General tab. The default repository is located in the *`BEA_HOME/user_projects/platform_domain_name`* directory such as:

`BEAHOME/user_projects/domains/workshop/ldrepository`

On each Managed Server, you need to configure the repository location by mounting (UNIX NFS) or mapping (Windows) logical drives to point to the configured repository root on the Administration Server. The directory must be shared and mapped to a virtual disk name. The path must be identical to the value specified in the `ldrepository` root directory field on the General tab in the Liquid Data node.

– On UNIX, use NFS to mount the exported directory in the respective *`BEA_HOME/user_projects/platform_domain_name`*. Make a shortcut of the mapped drive in the managed domain and name it the same as that used in the administration server domain.

– On Windows, map to the drive using the exact same drive letter and path (such as `R:\ldrepos`).

4. Further configure or reconfigure data sources, security, the repository, and other Liquid Data server settings, as needed, through WebLogic Administration Console, as described in the Liquid Data *Administration Guide*.

5. If you plan on running the domain as a Windows Service, modify the service startup script as described in "Modifying the Windows Service Startup Script to Support Liquid Data" on page 3-16.

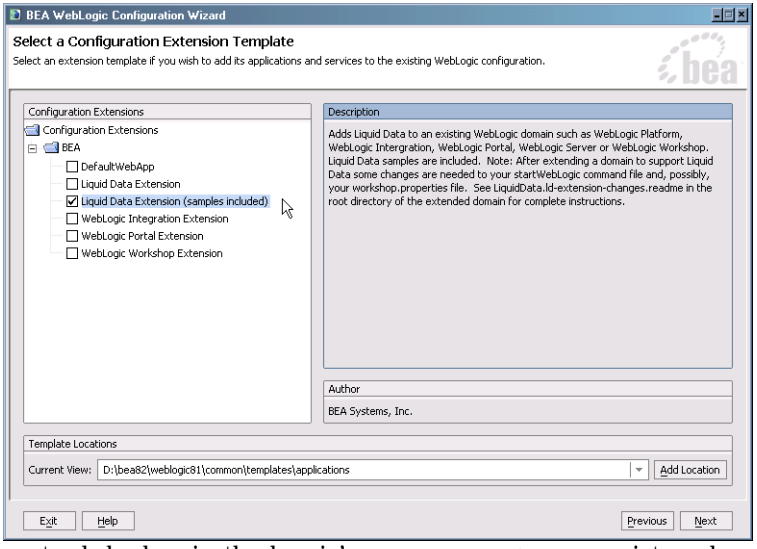
For detailed information on using the WebLogic Configuration Wizard as well as deployment concepts see "Overview of WebLogic Platform Configuration" in *Configuring* WebLogic Platform. Also see WebLogic Configuration Wizard on-line help.

Extending Liquid Data to an Existing WebLogic Domain

You can also use the WebLogic Configuration Wizard to import a Liquid Data configuration template into an existing WebLogic Platform domain.

Note: You cannot use the WebLogic Configuration Wizard to extend a clustered domain for Liquid Data + Samples.

Figure 3-2 Dialog Box Used to Extend an Existing Domain Configuration to Support Liquid Data



After you have extended a domain, the domain's `startWebLogic.cmd` script needs some minor editing. If you are extending a domain that includes WebLogic Workshop data from two versions of a workshop.properties file may need to be merged. Details are provided in "Required Changes to Server Files" on page 3-9.

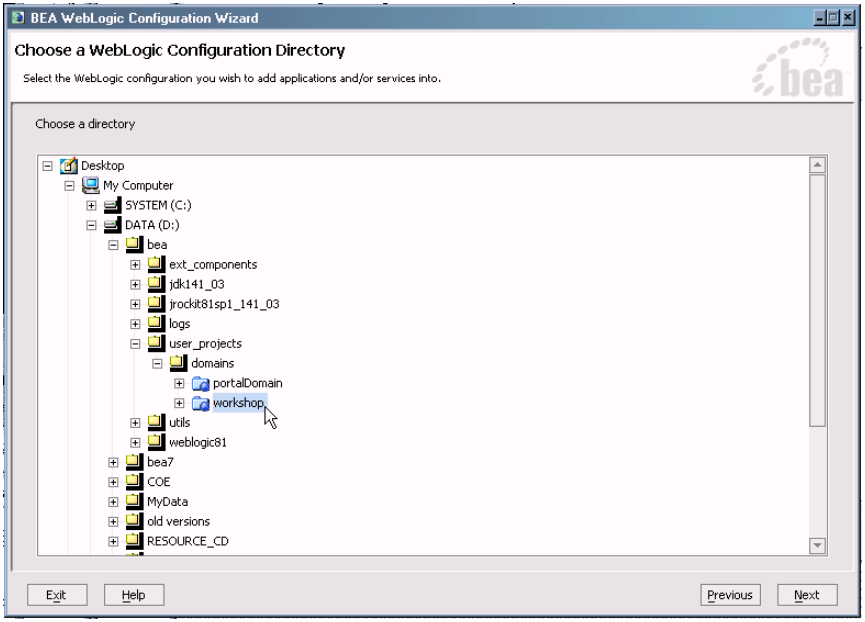
Here are the general steps required to extend an existing WebLogic Server domain:

1. Start the WebLogic Configuration Wizard and select the Extend an Existing WebLogic Domain option.

2. Use the WebLogic Configuration Wizard directory browser to locate the domain (in this case the domain is named `workshop`, see Figure 3-3) you wish to extend. Then click Next.

3. There are two ways to extend a domain for Liquid Data: with or without the Liquid Data sample domain included. Once you have checked the extension you want, click Next.

Figure 3-3 Selecting a Domain in the WebLogic Configuration Wizard



When extending a domain, you can accept default settings or customize the domain. Customization includes:

- Database (JDBC) options
- Messaging (JMS) options such as stores, topics, and queries
- Identifying target servers and clusters onto which applications, JMS component services, JDBC component services, and other services are deployed. (In a multi-module deployment the `ldconsole.jar` is only deployed on the Administration Server; all other modules are deployed on Managed Servers. See "Deployable Liquid Data Components" on page 2-2 for details.)
- Establishing application security, users, and groups

For detailed information on using the WebLogic Configuration Wizard as well as deployment concepts see "Overview of WebLogic Platform Configuration" in *Configuring* WebLogic Platform. See also WebLogic Configuration Wizard on-line Help.

Configure or reconfigure data sources, security, the repository, and other Liquid Data server settings, as needed, through WebLogic Administration Console, as described in the Liquid Data *Administration Guide*.

4. Edit command script files to support Liquid Data.

   The following table shows Liquid Data-enabled deployment options and identifies scripts that need to be changed for the particular option.

Figure 3-4 Command Files Requiring Editing When Extending a Domain for Liquid Data or Liquid Data + WebLogic Application Integration (WLAI)

Operating System	Process	Liquid Data Extension
Windows	Start WebLogic	• Start Administration Server `startWebLogic.cmd` • Start Managed Server `startManagedWebLogic.cmd` • Workshop Properties, if present `workshop.properties`
UNIX	Start WebLogic	• Start Administration Server `startWebLogic.sh` • Start Managed Server `startManagedWebLogic.sh` • Workshop Properties, if present `workshop.properties`

For example, if you have an existing WebLogic Workshop 8.1 domain and want to import a Liquid Data template, you would:

a. Run the WebLogic Configuration Wizard and extend the WebLogic Workshop domain for the Liquid Data option.

b. Modify the following command file:

`startWebLogic.cmd` (if an Administration Server) *or* `startManagedWebLogic.cmd` (if a Managed Server)

c. Start or restart the server(s).

5. If you are extending a clustered deployment you need to decide how your managed servers will access a shared `ldrepository` directory.

   – If the deployed Liquid Data repository seldom if ever changes, you can simply set the `ldrepository` directory structure in the Liquid Data node of the WebLogic Administration Console (Configuration —> General —> Repository Directory) to a specific location and then copy the `ldrepository` directory and its contents to the same exact location on each managed server.

     For example of you had a cluster with two managed servers your `ldrepository` directory might be located at:

     Managed Server No. 1:

     - `/opt/prod/bea_81/user_domains/ld_domain/ldrepository`

     Managed Server No. 2:

     - `/opt/prod/bea_81/user_domains/ld_domain/ldrepository`

     The advantage of this approach is that you will get better performance with a local repository on each machine.

   – Alternatively, you can change the `ldrepository` directory path so that it points to a shared volume to which all Managed Servers have access. On Windows, for example, you would point to a shared network drive mapped to a specific drive letter (such as `R:\ldrepos`).

   In both cases you set the Liquid Data repository path using the Liquid Data node of the WebLogic Administration Console as follows:

   a. Start or restart the Administration Console, logging in as appropriate.

   b. In the left pane, click on the Liquid Data node, then click the General tab. The default repository is located in the *`BEA_HOME/user_projects/platform_domain_name`* directory such as:

      `BEAHOME/user_projects/domains/workshop/ldrepository`

   On each Managed Server, you need to configure the repository location by mounting (UNIX NFS) or mapping (Windows) logical drives to point to the configured repository root on the Administration Server. The directory must be shared and mapped to a virtual disk name. The

path must be identical to the value specified in the `ldrepository` root directory field on the General tab in the Liquid Data node.

  - On UNIX, use NFS to mount the exported directory in the respective *`BEA_HOME/user_projects/platform_domain_name`*. Make a shortcut of the mapped drive in the managed domain and name it the same as that used in the administration server domain.
  - On Windows, map to the drive using the exact same drive letter and path (such as `R:\ldrepos`).

6. Further configure or reconfigure data sources, security, the repository, and other Liquid Data server settings, as needed, through WebLogic Administration Console, as described in the Liquid Data *Administration Guide*.
7. If you plan on running the domain as a Windows Service, modify the service startup script as described in "Modifying the Windows Service Startup Script to Support Liquid Data" on page 3-16.

Required Changes to Server Files

WebLogic Platform Servers are started by command scripts. These scripts have either `cmd` (Windows) or `sh` (UNIX) extensions. You need to make some minor editing changes to these scripts when deploying Liquid Data into an existing domain.

- Administration Servers are started by `startWebLogic` scripts
- Managed Servers are activated by `startManagedWebLogic` scripts

Changes required to server command scripts are described in the following sections:

- Changes to Domain Command Scripts Enabled for Liquid Data
  - Changes to Windows Start WebLogic Scripts
  - Changes to UNIX Start WebLogic Scripts

Note: Command files are located in the root directory of the your extended domain and contain the following "readme" file:

```
LiquidData.ld-extension-changes.readme
```

Note: Generated command files may vary somewhat from examples shown.

Changes to Domain Command Scripts Enabled for Liquid Data

This section describes changes needed in Windows or UNIX command script files to support domains extended to support Liquid Data but not WebLogic Application Integration.

Changes to Windows Start WebLogic Scripts

Edit `startWebLogic.cmd`, using these steps:

a. Add the following command lines after WL_HOME is defined:

```
set LD_HOME=%WL_HOME%\<LD_HOME>
call %LD_HOME%\setenv.cmd
```

where <*LD_HOME*> is the root Liquid Data directory.

Example:

```
set WL_HOME=D:\bea\weblogic81
set LD_HOME=%WL_HOME%\liquiddata
call %LD_HOME%\setenv.cmd
```

b. Add the following command to the class path:

```
%LDCOMMONCP%
```

Example:

```
@rem Call WebLogic Server
set
CLASSPATH=%WEBLOGIC_CLASSPATH%;%POINTBASE_CLASSPATH%;%JAVA_HOME%\jre\lib\rt.ja
r;%WL_HOME%\server\lib\webservices.jar;%CLASSPATH%;%LDCOMMONCP%
```

Changes to UNIX Start WebLogic Scripts

Edit the `startWebLogic.sh` file.

a. Add the following command lines after WL_HOME is defined:

```
LD_HOME=$WL_HOME/<LD_HOME>
.$LD_HOME%/setEnv.sh
```

where <*LD_HOME*> is the root Liquid Data directory.

Example:

```
WL_HOME=/bea/weblogic81

LD_HOME=$WL_HOME/liquiddata
. "$LD_HOME/setEnv.sh"
```

b. Insert the following line immediately before the startweblogic server command (and after other CLASSPATH definitions):

```
CLASSPATH=$CLASSPATH:$LDCOMMONCP
```

Example:

```
# Start WebLogic server

CLASSPATH="${WEBLOGIC_CLASSPATH}${CLASSPATHSEP}${POINTBASE_CLASSPATH}${CLASSPA
THSEP}${JAVA_HOME}/jre/lib/rt.jar${CLASSPATHSEP}${WL_HOME}/server/lib/webservi
ces.jar${CLASSPATHSEP}${CLASSPATH}"

export CLASSPATH

CLASSPATH=$CLASSPATH;$LDCOMMONCP
# supports Liquid Data extension

"$JAVA_HOME/bin/java" ${JAVA_VM} ${MEM_ARGS} ${JAVA_OPTIONS}
```

Changes to the workshop.properties file

After the Liquid Data extension is imported into a domain, it will add a file called `workshop.properties` into the domain. If a `workshop.properties` file existed before the Liquid Data extension was imported, the old file will be automatically backed up by the WebLogic Configuration Wizard to the filename `workshop.properties.bak`

If you find such a file in the root directory of the server you are extending, you need to merge that file with the new `workshop.properties` file. The order of the files being merged does not matter.

Modifying the Startup Script To Set the Administrative User Name

If you use the Domain Configuration Wizard to create a Liquid Data production domain, or if you have a domain that does not contain the `boot.properties` file at the root level, you must modify the startup script to specify the name of an administrative user (a user who is a member of the WebLogic `Administrators` group). If the domain does not have the administrative username set, an error will occur on server startup.

The startup script for Liquid Data domains created with the Domain Configuration Wizard contains the following code fragment:

```
if "%WLS_PRODUCTION_MODE%"=="true" (
       set JAVA_OPTIONS=%JAVA_OPTIONS% -Dcom.bea.ld.adminusername=
       )
```

Note: In order for the startup script to correctly run, an administrator's user name must be supplied.

Perform the following steps to update the startup scripts for production domains or domains that do not have a `boot.properties` file:

1. Open the startup script for your domain in a text editor. The startup script is in the root level of the domain and is named `startWebLogic.cmd` for Windows systems and `startWebLogic.sh` for UNIX systems.
2. Find the lines shown above in the startup script.
3. Enter the name of a user who is a member of the Administrators group following the terminating equals (=) sign. For example, if an administrative user is named `system_admin`, change:

```
JAVA_OPTIONS="%JAVA_OPTIONS% -Dcom.bea.ld.adminusername=system_admin"
```

   to the following (emphasis added):

```
JAVA_OPTIONS="%JAVA_OPTIONS% -Dcom.bea.ld.adminusername=system_admin"
```

4. Save the file.
5. Start the domain.

Copying a Liquid Data Configuration From Server to Server

There are two ways to copy a Liquid Data server configuration information from one server to another, such as from a development environment to a production environment.

1. You can use the WebLogic Configuration Template Builder to create a domain template and then use the WebLogic Configuration Wizard to deploy it. To do this some detailed understanding of the composition of a domain is needed.

   You can access the Configuration Template Builder from the following locations:

   Windows:

   ```
   <wl_home>\common\bin\config_builder.cmd
   ```

   UNIX:

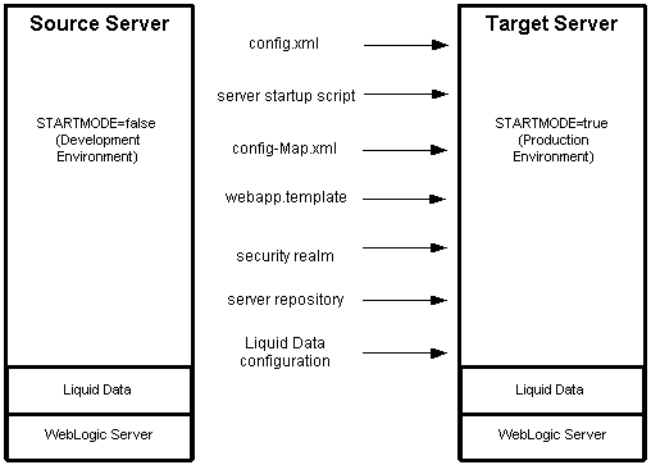
   ```
   <wl_home>/common/bin/config_builder.sh
   ```

For more information see Creating Configuration Templates Using the Weblogic Configuration Template Builder.

2. You can also can transfer settings already configured in the development environment.

The following illustration shows the Liquid Data components that you need to transfer to the production server:

Figure 3-5 Migrating From a Development to a Production Server



To migrate from a development environment to a production environment, you need to migrate the following configuration information:

Table 3-6 Liquid Data Configuration Information that Can Be Migrated

Item	Description
`config.xml` file	Contains the WebLogic Server configuration, including the domain configuration and, if applicable, the JDBC database configuration. Open the `config.xml` file on the target server and edit any machine-specific parameters.
startup script (`startWebLogic.cmd` or `startWebLogic.sh`)	Contains the Liquid Data server startup script. Open the startup script on the target server and edit any machine-specific parameters. Also, for a production environment, change the `STARTMODE` variable to `true`.
`configMap.xml` and `webapp.template` files	Copy these files from `<WL_HOME>/<LIQUIDDATA>/server/lib` on the source server to the corresponding directory on the target server. (Not necessary if the target server already has Liquid Data enabled.)

Table 3-6 Liquid Data Configuration Information that Can Be Migrated (Continued)

Item	Description
security realm	Replicate the security realm configuration on the production server. The Liquid Data security configuration includes groups, users, and roles. How you copy the configuration depends on the type of compatibility security realm that you have set up according to the instructions in "Defining a Compatibility Security Realm" in "Implementing Security" in the Liquid Data *Administration Guide*. (Not necessary if the target server already has Liquid Data enabled.)
server repository	On the development server, create a compressed image of the complete server repository, including all sub-folders, using a TAR or ZIP file for the Unix and Windows platforms, respectively. Copy this compressed file to the production server and expand it. On the production server, start the WebLogic Administration Console and configure the repository root directory to be the directory where the source repository was unzipped to.
Liquid Data configuration settings	On the development server, export the Liquid Data configuration information to an export file. On the production server, import the contents of the export file. For instructions, see "Importing and Exporting Liquid Data Configurations" in the Liquid Data *Administration Guide*.
Web services and other live applications	If there are any "live" applications in the development server such as an `.EAR` file for a web service, you should move a copy of the file into the application directory after the target server is up and running. This needs to be done before changing the Start mode to True.

Configuring Remote Start for a Liquid Data-Enabled Managed Server

If you create a clustered Liquid Data domain or extend an existing clustered domain to support Liquid Data and you want to be able to remotely start your managed server(s) from the WebLogic Administration Console, you will need to insert instructions copied from the Administration Server's executed startWebLogic script to each Managed Server's configuration instructions.

You can use the following steps to accomplish this for each Managed Server in your cluster:

1. Start your Administration Server using the startWebLogic command script. For example:

   ```
   <user_domain>/<domain>/startWebLogic.cmd
   ```

where *user_domain>/<domain>* is the root directory of your WebLogic domain (for example, d:\user_domains\mydomain).

As you are copying sections of the executed script you may find it easiest to create a text file containing the output of the startWebLogic process. On Windows you can create such a file using:

```
startWebLogic >> scriptoutput.txt
```

2. Edit the output, concatenating the line beginning CLASSPATH= with the line beginning PATH=
3. Copy the resulting string (without any return codes) into the Class Path field in WebLogic Administration Console Configuration —> Remote Start page. This override the default startup classpath of your managed server.
4. Copy the arguments used in starting your server into the Arguments field in WebLogic Administration Console Configuration —> Remote Start page.

   The server startup line would being with something similar to:

   ```
   Starting WLS with line:
   d:\b11\JDK142\bin\java <server arguments>
   ```

   and the arguments to be copied into the field would include everything following the invocation:java

   This override the default startup arguments for your Managed Server.
5. Start your Node Manager, located at <*BEA_HOME*>/server/bin

   For example:

   ```
   d:\bea_81_sp3\weblogic81\server\bin\startNodeManager.cmd
   ```
6. Test the results by starting or restarting each Managed Server.

   You can locate your server by name under the Servers section of your domain in the WebLogic Administration Console. Once selected, you can use the Control tab to start or restart your Managed Server.

See Overview of Node Manager at http://e-docs.bea.com/wls/docs81/adminguide/nodemgr.html for additional information.

Modifying the Windows Service Startup Script to Support Liquid Data

If you create a Liquid Data domain and want to run it as a Windows Service, you must modify the startup script for the Window Service to include needed Liquid Data resources in the classpath.

If you have created your domain in the WebLogic Configuration Wizard and selected the Run as a Service option, then a file named `installService.cmd` will be in your domain directory. You will need to edit that file.

Alternatively you can create a script which invokes the Windows service command file located in `<WL_HOME>/server/bin` directory. If that is how your will start your Windows service then you will need to edit the `installSvc.cmd` file located in that directory.

The editing instructions for `installSvc.cmd` or `installService.cmd` are the same.

1. Open the appropriate file in a text editor to add Liquid Data resources to the classpath.
   - If you selected the Run as a Service option in the Configuration Wizard when you created the domain open:

     <user_domain>/<domain>`/installSvc.cmd`

     where *user_domain>/<domain>* is the root directory of your WebLogic domain (for example, `d:\user_domains\mydomain`).
   - Otherwise open:

     `<WL_HOME>/server/bin/installService.cmd`

     where *WL_HOME* is the directory in which WebLogic is installed (for example, `d:\bea\weblogic81`).
2. Modify the file by changing the line setting the class path from:

```
set CLASSPATH=%WEBLOGIC_CLASSPATH%;%CLASSPATH%
```

to:

```
call "%WL_HOME%\liquiddata\setenv.cmd
set CLASSPATH=%WEBLOGIC_CLASSPATH%;%CLASSPATH%;%LDCOMMONCP%
```

3. Locate the following code in the file:

```
if "%PRODUCTION_MODE%"=="true" (
       set JAVA_OPTIONS=%JAVA_OPTIONS% -Dcom.bea.ld.adminusername=
       )
```

In order for the startup script to correctly run, an administrator's user name must be supplied.

4. Enter the name of a user who is a member of the Administrators group following the terminating equals (=) sign. For example, if an administrative user is named `system_admin`, change:

```
JAVA_OPTIONS="%JAVA_OPTIONS% -Dcom.bea.ld.adminusername=system_admin"
```

   to the following (emphasis added):

```
JAVA_OPTIONS="%JAVA_OPTIONS% -Dcom.bea.ld.adminusername=system_admin"
```

5. Save your file (either `installSvc.cmd` or `installService.cmd`).

6. Perform the steps in the WebLogic Server documentation for running WebLogic Server as a Windows Service:

   http://e-docs.bea.com/wls/docs81/adminguide/winservice.html

7. Startup and test the domain running as a Windows Service.

CHAPTER 4

Troubleshooting a Deployment

This topic provides information to help you troubleshoot a BEA Liquid Data for WebLogic deployment. It includes the following sections:

- Troubleshooting Resources
- Troubleshooting Out of Memory Exceptions

This section provides general troubleshooting tips. For information about known limitations in specific Liquid Data releases, see the Liquid Data Release Notes.

Troubleshooting Resources

The following sections describe resources that you can use to troubleshoot a Liquid Data deployment:

- Liquid Data Resources
- WebLogic Server Resources
- WebLogic Portal Resources
- WebLogic Workshop Resources
- BEA Developer Center

Liquid Data Resources

Liquid Data provides log entries and release notes for troubleshooting resources.

Log Entries

Review the log entries in the domain log for errors. Liquid Data records status, problem, and performance information in the domain log. For more information, see "Monitoring the Server Log" in "Monitoring the Server" in the Liquid Data *Administration Guide*.

Release Notes

The Liquid Data Release Notes provide information about known limitations and workarounds for the version of Liquid Data that you are using.

Product Documentation

The Liquid Data product documentation provides detailed information about all aspects of the Liquid Data product. For more information, see the documentation CD that comes with the Liquid Data package or go to the Liquid Data Documentation page.

WebLogic Server Resources

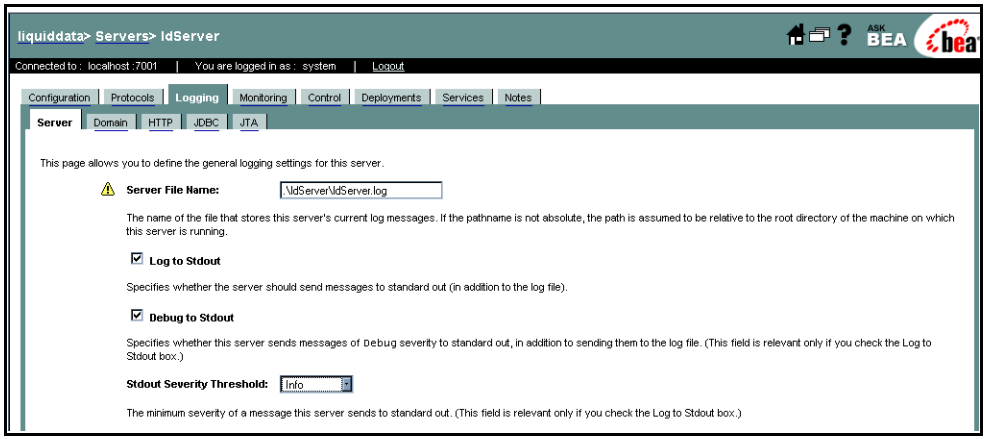
The BEA WebLogic Server server log contains valuable information about startup and run-time events that can help you troubleshooting a Liquid Data deployment. You can configure the amount of information that the WebLogic Server saves in the log file.

To configure the WebLogic Server server log:

1. On the Liquid Data server, start the WebLogic Server, then log into the WebLogic Administration Console.
2. In the left pane of the Administration Console, click the name of the Liquid Data server for which you want to configure the log.
3. Click the Logging tab.

   The Administration Console displays the General tab showing the current log settings.

Figure 4-1 WebLogic Server Log Configuration Page (Partial) Set to Maximize Logging Information



4. Change the log settings as needed.

   If you want the WebLogic Server to save the maximum amount of information to the log, select Log to Sdout, Debug to Stdout, and select a severity level of Info.

5. Click Apply.

Note: For debugging purposes, you can also click the Debugging tab, select Log Remote Exceptions and Instrument Stack Traces, and click Apply.

In a clustered environment, the domain log provides an aggregated view of log events for all servers in the cluster. For detailed information about WebLogic Server logs, see "Server Log" in the *WebLogic Server Administration Console Online Help*.

WebLogic Portal Resources

BEA WebLogic Portal uses the WebLogic Server logging mechanism to save messages to the WebLogic Server log. WebLogic Portal applications can control logging levels using classes and interfaces in the WebLogic Portal API, as described in the WebLogic Portal Javadoc.

WebLogic Workshop Resources

BEAWebLogic Workshop uses the log4j Java logging facility developed by the Jakarta Project of the Apache Foundation. You can learn more about log4j at The Log4j Project.

You configure logging in WebLogic Workshop by changing settings in the `workshopLogCfg.xml` file, which by default is located in *`BEA_HOME`*`/weblogic81/common/lib`. For more information, see

"workshopLogCfg.xml Configuration File" in the *WebLogic Workshop Reference* on-line documentation.

BEA Developer Center

The BEA Developer Center provides a range of technical resources, including newsgroups on technical topics, such as installation, clustering, JDBC, EJBs, servlets, and JSPs. For more information, visit the BEA Developer Center at the following URL:

http://support.bea.com/index.jsp

Troubleshooting Out of Memory Exceptions

This topic describes how to handle out of memory exceptions that can occur in a deployment. The following table provides a list of possible causes and associated fixes.

Table 4-2 Causes and Solutions for Out of Memory Exceptions

Possible Cause	Problem Description	Solution
Lack of Proper Hints	Lack of proper or correct hints.	Specify proper hints, as described in "Optimizing Queries" in *Building Queries and Data Views*. Alternatively, redesign the query.
Query Plan Not Optimized	• Lack of proper or correct hints. • Possible limitations in Liquid Data Server implementation.	Specify proper hints. Redesign the query.

Table 4-2 Causes and Solutions for Out of Memory Exceptions (Continued)

Possible Cause	Problem Description	Solution
Large Result Set	Query returns a very large result set. For example, a query retrieves all the customers in the database.	• Increase the heap size from the default (256MB) to as high as possible. • Specify hints. For example, for RDBMS data sources, use the `merge` hint, if applicable, although using the merge hint might slow query performance. • Use disk swapping, which improves system availability but might slow system performance.
Large Intermediate Result Set	Query returns a very large intermediate result set. For example, an analytical query, or a query that processes a large XML document, which Liquid Data loads entirely into memory before any query processing.	• Increase the heap size from the default (256MB) to as high as possible. • Use disk swapping, which improves system availability but might slow system performance.

CHAPTER 5

Tuning Liquid Data Performance

This chapter describes how to tune performance in a BEA Liquid Data for WebLogic deployment. It includes the following sections:

- Where to Begin
- Liquid Data Performance Factors
- Monitoring Liquid Data Performance

In a production environment, Liquid Data performance is generally measured by the speed with which queries are processed and results are returned. This topic describes general guidelines for performance and, where possible, provides specific guidelines for tuning a Liquid Data deployment.

Where to Begin

This topic describes where to begin tuning a Liquid Data deployment. It contains the following sections:

Checking Your System Configuration

Before you begin to investigate how to tune Liquid Data performance, make sure that the system on which Liquid Data runs is reasonably configured.

- **System platform configuration.** Do you have sufficient CPU, memory, and disk space resources? A properly tuned heap size? The right Java Virtual Machine (JVM)? Reasonable network speed? For more information, see "Platform Performance Factors" on page 5-9.
- **Data sources configuration.** Are data sources properly tuned? For example, if the deployment uses relational database data sources, do the tables in those data sources have adequate indexes? Is the JDBC connection pool size reasonably configured? If the deployment uses Web Services data sources, is the Web Service available and does it provide a reasonable response time? For more information, see "Data Source Performance Factors" on page 5-6.

Tuning Queries

In addition to the system configuration, Liquid Data performance is greatly affected by the query design, as described in "Query Performance Factors" on page 5-3. Therefore, make sure that the queries running on the Liquid Data server are reasonably designed:

- Does each query follow the appropriate query design pattern, as described in "Design Patterns" in "Designing Queries" in *Building Queries and Data Views*?
- Does each query provide the appropriate hints, as described in "Optimizing Queries" in *Building Queries and Data Views*?
- Does each query have the appropriate scope, as described in "Understanding Scope in Basic and Advanced Views" in "Designing Queries" in *Building Queries and Data Views*?
- Does any query produce a large intermediate result set or final result set? For more information, see "Query Performance Factors" on page 5-3.

- Have you debugged the query using a verbose logging mode (info) and reviewed the trace? For more information, see "Monitoring Liquid Data Performance" on page 5-14.

Liquid Data Performance Factors

Many factors influence overall Liquid Data performance. Certain factors, such as query design, are within Liquid Data's scope, while other factors, such as data source processing speed, are outside Liquid Data's area. This topic identifies the main factors that can affect Liquid Data performance. It includes the following sections:

- Query Performance Factors
- Data Source Performance Factors
- Platform Performance Factors

Query Performance Factors

Liquid Data performance depends on the way queries are designed and configured for execution. The following query factors affect Liquid Data performance:

Table 5-1 Query Performance Factors

Factor	Description
Query complexity	Some query operations are more resource intensive than others. • Simple queries perform basic operations, such as retrieving a customer list or an employee's profile. • Analytical queries, such as complex joins or aggregates, perform interim processing steps that produce intermediate result sets. **Recommendations**: Because analytical queries generally consume more memory and CPU resources than simple queries, see the sections later in this table on caching and large result sets.

Table 5-1 Query Performance Factors (Continued)

Factor	Description
Query type	The type of query (stored or ad hoc) affects performance: • For stored queries, the query is compiled only once and query execution plan is cached. In addition, the query result can be cached for faster retrieval if the query is executed subsequently with the same parameters. For more information, see "Configuring the Query Results Cache" in the Liquid Data *Administration Guide*. • For ad hoc queries, the query execution plans are compiled once and cached, but the query results are not cached. For certain queries, the time to compile the query might take much longer then the time to execute. **Recommendation**: Use stored queries in a production system.
Design pattern used (query hints)	When building queries, the appropriate design patterns must be used to ensure the fastest possible execution speed. For example, for joins, you should use a query hint to supply as much information to the execution engine as possible about the amount of data to search when processing a query. **Recommendation**: Use hints, if applicable. Hints are particularly significant with large data sources. If you are using the Data View Builder, design patterns for target schema are also very important. For more information, see "Understanding Query Design Patterns" in "Designing Queries" and "Optimizing Queries" in *Building Queries and Data Views*.
Caching	Caching improves performance for stored queries. Liquid Data supports two types of caching: • Query plan cache, which is always enabled, allows Liquid Data to retrieve the query plan for a repeated query from cache rather than regenerating the query plan. • Query results cache allows Liquid Data to retrieve the results for a repeated query from in-memory cache (rather than running the query against the data source each time). Stored query results that are configured to expire sooner must be executed against the data source(s) more frequently. **Recommendations**: • Always use stored queries in a production system. • Use query results caching, if applicable. For a complex query, or for a query that retrieves data from slower data sources, caching the query result provides a substantial performance gain. Once the query is cached, the query execution time is the fixed cost of retrieving the results cache from the database plus returning it as an XML document. For more information, see "Configuring the Query Results Cache" in the Liquid Data *Administration Guide*.

Table 5-1 Query Performance Factors (Continued)

Factor	Description
Size of intermediate or final results returned and memory usage	Generally the larger the result size (final or intermediate results), the longer it takes to retrieve and process the results. **Recommendations**: • Queries should be designed to retrieve only the required results. Restrict the number of elements in the target schema (project). Specify strict conditions that limit the number of rows returned in the result set. • Increase the heap size to as large as the system can allow. In general, increasing the heap size increases performance. If increasing the heap size solves the out-of-memory problem, then no further action is required. • For relational database data sources, use the `{--! merge !-- }` query hint, if applicable. The effect is similar to a database merge join. Using this technique has a space/performance trade-off: a merge hint provides better memory usage but it might also slow performance if extra sorting is required to perform the merge join. • For queries that perform joins on multiple data sources, specify the sequence of data sources in ascending order by increasing size: the smallest resource should appear in the first FOR loop, and the largest resource should appear in the last one. For more information, see "Source Order Optimization" in "Optimizing Queries" in *Building Queries and Data Views*. • Use the disk swapping option, which allows Liquid Data to store intermediate results on disk when the results exceed the `MEM_SORT_BUF` size specified in the server startup script (`startWebLogic.cmd` on Windows or `startWebLogic.sh` on Unix). The default setting for `MEM_SORT_BUF` size is 50MB. The recommended size is less than one third (1/3) of the maximum memory size defined in `MEM_ARGS` in the server startup script. To configure disk swapping, see "Configuring Liquid Data Server Settings" in the *Administration Guide*. **Note:** Liquid Data queries do not retrieve binary large object (BLOB) data from relational databases. See "Supported Data Types" in *XQuery Reference Guide*.
Number of concurrent queries	The higher the number of concurrent queries, the slower the performance, particularly during peak loads. Performance improves through the use of additional CPUs and WebLogic Server clusters, as described in "Clustered Deployments" on page 2-5, and with tuning the thread pool, as described in "Using the WebLogic Administration Console to Monitor Performance" on page 5-15.

Data Source Performance Factors

In general, Liquid Data performance depends on the speed at which the data source host system is able to process query requests and return results.

Performance Factors for All Data Sources

The following data source factors affect Liquid Data performance:

Table 5-2 Data Source Performance Factors

Factor	Description
Data source type	Some types of data sources offer higher performance (such as relational databases) than other types (such as application integrations or Web Services). For more information, see Table 5-3, "Performance Factors for Data Source Types," on page 5-8.
Data source size	The size of the data source affects performance. In general, the larger the data source, the longer it takes to retrieve the query results. For example, a large XML document takes longer to process than a small XML document. For relational databases, indexing can very substantially improve performance, particularly for large databases.
Number of data sources	For queries that access multiple data sources, data is retrieved from each data source in sequence, one data source at a time. This is true for all data source types except application views, Web Services, and custom functions (which are processed asynchronously). For application views, Web Services, and custom functions you can configure the maximum number of connections or the maximum number of concurrent threads to be used. If queries use Web Services, application views, or custom functions extensively, then consider tuning this setting.

Table 5-2 Data Source Performance Factors (Continued)

Factor	Description
Data source performance and availability	• A query fails if a required data source is unavailable during query execution due to server failure, insufficient available connections, failed authentication, or other factors. • Performance delays result if the server hosting the data source is congested. • Faster hardware (storage, memory, and CPU throughput) for the data source host machine generally provides higher performance, particularly for larger data sources. • Except for any XML files stored locally on the host system, all data sources are remote. Therefore, network connection availability and speed affects how quickly the query results are returned. In addition to network capacity and throughput speeds, the number of hops between nodes can greatly affect performance. Secure connections, such as SSL (Secure Sockets Layer) increase security but slow performance. Network speed is not a factor with local data sources, such as XML files stored locally on the host system. However, network speed is a factor with XML files stored remotely.
Transaction isolation level (relational databases only)	For relational databases, the transaction isolation level setting can affect query performance: • For databases containing static or read-only data, in general, use the default setting (`TRANSACTION_READ_COMMITTED`). • For databases containing dynamic or updateable data, use the transaction isolation level that supports the degree of concurrency required. However, increased concurrency can result in slower query performance. Using a transaction level of `TRANSACTION_SERIALIZABLE`, the highest level of concurrency, is more likely to reduce performance than using a lower level of concurrency, such as `TRANSACTION_REPEATABLE_READ` or `TRANSACTION_READ_COMMITTED`. For more information about configuring the transaction isolation level for a relational database, see "Creating a Relational Database Data Source Description" in "Configuring Access to Relational Databases" in the Liquid Data *Administration Guide*.

Performance Factors for Data Source Types

The following table describes the most important performance factors for each supported data source type:

Table 5-3 Performance Factors for Data Source Types

Type	Important Performance Factor(s)
Relational databases	• A fast host machine is especially important for large databases. • An optimized database design and a highly tuned configuration substantially improves performance. For example, indexing improves performance, particularly for large databases. For more information, see your database vendor's documentation. • The JDBC driver configuration can affect query performance, and JDBC connection pool settings must be properly tuned. For more information, see "JDBC" in the BEA WebLogic Server Administration Console Online Help. • The JDBC transaction isolation level can affect query performance. For more information, see the discussion of transaction isolation levels in Table 5-2, "Data Source Performance Factors," on page 5-6.
XML files	• Size of file in generally predictive of performance.
Web Services	• Query performance depends primarily on the Web Service implementation and the speed at which the Web Service returns results, as well as the network connection speed. • Liquid Data runs query requests on Web Services asynchronously. If a query uses multiple data sources, the query can continue processing while waiting for a response from a Web Service. For Web Services, therefore, performance also depends on the Maximum Threads setting specified on the General tab in the Liquid Data node in the Administration Console, as described in "Configuring Liquid Data Server Settings" in the Liquid Data *Administration Guide*.
Application views	• Query performance depends on the performance of the EIS and its associated adapter. • Liquid Data runs query requests on application views asynchronously. If a query uses multiple data sources, the query can continue processing while waiting for a response from an application view. For application views, therefore, performance also depends on the Maximum Threads setting specified on the General tab in the Liquid Data node in the WebLogic Administration Console, as described in "Configuring Liquid Data Server Settings" in the Liquid Data *Administration Guide*.
Data views	• Complexity of the underlying query, as described in "Query Performance Factors" on page 5-3. • The data source performance factors described in "Performance Factors for All Data Sources" on page 5-6.

Platform Performance Factors

This section describes performance factors associated with the Liquid Data server, including the host server hardware, clustering Liquid Data servers, tuning threads, tuning WebLogic Server, and tuning WebLogic Integration. The most important factor is running Liquid Data on a very fast server machine with the maximum amount of available memory. For general information about platform performance, see "Tuning Hardware, Operating System, and Network Performance" in *BEA WebLogic Server Performance and Tuning*.

This section describes the following platform performance factors:

- General Platform Performance Factors
- WebLogic Integration Performance Factors
- Liquid Data Host Server Machine
- WebLogic Integration Performance Factors

General Platform Performance Factors

The following general performance factors are associated with a Liquid Data deployment:

Table 5-4 Server Hardware Performance Factors

Factor	Description
Network connection speed	For remote resources such as data sources, the speed and capacity of the network connection is an important factor. In addition to network capacity and throughput speeds, the number of hops between nodes can greatly affect performance. Secure connections, such as SSL (Secure Sockets Layer) increase security but slow performance.
Distribution of resources across servers	Performance is greatly affected by the way in which Liquid Data, WebLogic Server, and other WebLogic Platform resources are distributed across servers. For example: • If a Liquid Data deployment uses application views as data sources, it is generally optimal to run Liquid Data and Application Integration on separate server machines, as described in "Multi-Node Deployments" on page 2-4. • Liquid Data is built on the scalable WebLogic Server platform. In deployments that support a high volume of concurrent query requests, you can increase system performance by deploying on a WebLogic Server cluster—a group of WebLogic Servers that are managed as a single unit and distribute the load for processing query requests. For more information, see "Clustered Deployments" on page 2-5 and "Designing Deployments" on page 2-3.

WebLogic Server Performance Factors

Liquid Data performance is affected by WebLogic Server performance. The WebLogic Server documentation provides a detailed suggestions for monitoring and tuning run-time performance. For detailed information, see BEA WebLogic Server Performance and Tuning in the WebLogic Server documentation.

The following table provides a summary of WebLogic Server tuning factor:

Table 5-5 Summary of WebLogic Server Performance Factors

Component	Tunable Performance Factor(s)
Hardware Resources	• WebLogic Server Platform Supported Configurations http://e-docs.bea.com/wls/certifications/certifications/index.html • CPU—maximize speed and throughput • Memory—maximize capacity and speed • Network resources—maximum bandwidth and speed • Disk I/O and controllers—maximize disk speed and capacity • Number of machines—increase as needed
Operating System	• Configurable file descriptor limits • Memory allocation for user processes • Configurable TCP tuning parameters • Configurable settings for the threading model
Network Resources	• Network hardware and software • Network bandwidth • LAN infrastructure
Java Virtual Machine (JVM)	• JVM vendor and version • JVM heap size and garbage collection • Generational garbage collection • Mixed client/server JVMs • Unix threading models • Just-in-time (JIT) JVMs

Table 5-5 Summary of WebLogic Server Performance Factors (Continued)

Component	Tunable Performance Factor(s)
WebLogic Server	• config.xml file parameters • weblogic-ejb-jar.xml parameters • WebLogic Server startup parameters • Java compiler • WebLogic Server clusters • JDBC driver and JDBC connection pool
WebLogic Server Applications	• Performance analysis tools • JDBC application tuning • Session persistence • Minimizing sessions • Execute queues and thread usage

Liquid Data Host Server Machine

Faster hardware (storage, memory, and CPU throughput), large capacity storage (for caching and disk swapping), and for the Liquid Data server host machine generally provides higher performance. The following performance factors are associated with the host server machine:

Table 5-6 Liquid Data Host Server Machine Performance Factors

Factor	Description
CPU utilization	Optimal utilization is up to 80%.
Storage utilization	Machine should have sufficient available workspace for disk swapping and other storage operations. For recommendations, see "Installation Prerequisites" in "Preparing to Install WebLogic Server" in the WebLogic Server *Installation Guide*.

Table 5-6 Liquid Data Host Server Machine Performance Factors (Continued)

Factor	Description
Memory utilization	• The default memory size is 256MB, but it is recommended that you add as much memory as the server machine can support. Memory size is one of the most significant factors affecting Liquid Data performance. • If Liquid Data uses application views or Web Services as data sources, the Maximum Threads setting specified on the General tab in the Liquid Data node in the Administration Console determines the number of threads available for asynchronous requests to application views and Web Services. For more information, see "Configuring Liquid Data Server Settings" in the Liquid Data *Administration Guide*.
Thread pools	• The pool size for execute threads should be optimized. Liquid Data uses WebLogic Server's sophisticated multi-threading capabilities to process concurrent query requests more efficiently. By default, WebLogic Server uses 15 threads for the thread pool size. For more information, see "Tuning WebLogic Server" in *WebLogic Server Performance and Tuning*. • You need to tune the thread pool size according to the characteristics of the query request load on the Liquid Data server. Too many or too few threads can impede performance. In general, increase the number of threads if CPU utilization for the workload is low, and decrease the number of threads if CPU utilization exceeds 80%. You may need to experiment by repeatedly changing the maximum number of threads until you determine the optimum setting for your deployment.

WebLogic Integration Performance Factors

If Liquid Data is deployed with WebLogic Integration, then WebLogic Integration performance might affect Liquid Data performance, depending on how the two components interact. The WebLogic Integration documentation provides a detailed suggestions for monitoring and tuning run-time performance. For detailed information, see "Understanding WebLogic Integration Clusters" in *Deploying WebLogic Integration Solutions*.

The following table provides a summary of tuning factors, which are described in detail in the WebLogic Integration documentation.

Table 5-7 Summary of WebLogic Integration Performance Factors

Component	Tunable Performance Factor(s)
Business Process Management	• Event listener message-driven bean for event-driven workflows
Application Integration	• application view bean for synchronous service invocations • thread pool size of the asynchronous request processor for asynchronous service invocations • J2EE-CA resource pool size for each adapter
B2B integration	There are no primary resources that can be tuned
WebLogic Server	• EJB pool and cache sizes • JDBC connection pool sizes • Execution thread pool size • Resource connection pools for J2EE Connector Architecture adapters • Large message support for B2B
Java Virtual Machine (JVM)	• JVM vendor and version • JVM heap size • Garbage collection control (Hotspot JVM)
Hardware Resources	• CPU—maximize speed and throughput • Memory—maximize capacity and speed • Network resources—maximum bandwidth and speed • Disk I/O and controllers—maximize disk speed and capacity • Number of machines—increase as needed

Table 5-7 Summary of WebLogic Integration Performance Factors (Continued)

Component	Tunable Performance Factor(s)
Operating System	• Configurable file descriptor limits • Memory allocation for user processes • Configurable TCP tuning parameters • Configurable settings for the threading model
JDBC Databases	• Number of concurrently opened cursors • Disk I/O optimization • Database sizing and organization of table spaces • Checkpointing • Database compatibility • Database monitoring

Monitoring Liquid Data Performance

This section describes how to monitor Liquid Data performance. It includes the following sections:

- Monitoring Guidelines
- Using the WebLogic Administration Console to Monitor Performance

For detailed information about monitoring and tuning resources for the Liquid Data server, see "Tuning WebLogic Server" in WebLogic Server Performance and Tuning.

Monitoring Guidelines

When monitoring Liquid Data performance, consider the following guidelines:

- Try simulating workloads and monitor performance at different load levels—peaks, lulls, and mid-range volumes—to determine optimal overall settings.
- At a minimum, monitor server memory, threads, and CPU usage at these various load levels.
- Characterize the query request workload in the production environment, determining which queries are used and how often they are requested. For example, you might learn that five queries represent 80% of the workload and 10 queries represent the remaining 20%.
- Determine whether this workload involves the execution of small queries many times or large queries a few times and configure the thread pool accordingly.

- Determine whether clustering is appropriate for your deployment. For more information, see "Clustered Deployments" on page 2-5.
- Review your domain's server log for certain performance information. Liquid Data records the time to generate the query plan, compile the query plan, and execute the query. For relational databases, Liquid Data also records the SQL statement it generated for the query.

Using the WebLogic Administration Console to Monitor Performance

You can use the WebLogic Administration Console to monitor performance on a Liquid Data server, including the following areas:

- Active execute queues—As work enters a WebLogic Server, it is placed in an *execute queue* and then assigned to a thread in which the work is processed. The size of the execute pool determines the number of threads that can be running concurrently on an execute queue. Threads consume resources, so it is important that you tune the number of threads to optimize performance, taking care not to slow performance by increasing the value unnecessarily. By default, WebLogic Server uses 15 threads for the thread pool size.
- JDBC connection pools—If queries in your deployment use relational databases for data sources, you should monitor any active JDBC connection pools for performance. A JDBC connection pool contains a group of JDBC connections. At run-time, when a query that uses an RDBMS data source is executed, the query borrows a connection from the connection pool, uses it, then returns it to the connection pool by closing it. The size of the JDBC connection pool determines the number of JDBC connections that can be used concurrently. Connections consume resources, so it is important that you tune the number of connections to optimize performance, taking care not to slow performance by increasing the value unnecessarily.

For detailed information about using the WebLogic Administration Console to monitor server performance, see "Top Tuning Recommendations for WebLogic Server" in BEA WebLogic Server Performance and Tuning.

Index

O

P

Q

R

S

T

W

X