BEA WebLogic Portal™®

Database Administration Guide

Version 8.1 Service Pack 5
Document Revised: October 2005

Copyright

Copyright © 2004-2005 BEA Systems, Inc. All Rights Reserved.

Restricted Rights Legend

This software and documentation is subject to and made available only pursuant to the terms of the BEA Systems License Agreement and may be used or copied only in accordance with the terms of that agreement. It is against the law to copy the software except as specifically allowed in the agreement. This document may not, in whole or in part, be copied, photocopied, reproduced, translated, or reduced to any electronic medium or machine readable form without prior consent, in writing, from BEA Systems, Inc.

Use, duplication or disclosure by the U.S. Government is subject to restrictions set forth in the BEA Systems License Agreement and in subparagraph (c)(1) of the Commercial Computer Software-Restricted Rights Clause at FAR 52.227-19; subparagraph (c)(1)(ii) of the Rights in Technical Data and Computer Software clause at DFARS 252.227-7013, subparagraph (d) of the Commercial Computer Software--Licensing clause at NASA FAR supplement 16-52.227-86; or their equivalent.

Information in this document is subject to change without notice and does not represent a commitment on the part of BEA Systems. THE SOFTWARE AND DOCUMENTATION ARE PROVIDED “AS IS” WITHOUT WARRANTY OF ANY KIND INCLUDING WITHOUT LIMITATION, ANY WARRANTY OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. FURTHER, BEA Systems DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE SOFTWARE OR WRITTEN MATERIAL IN TERMS OF CORRECTNESS, ACCURACY, RELIABILITY, OR OTHERWISE.

Trademarks or Service Marks

BEA, BEA WebLogic Server, Jolt, Tuxedo, and WebLogic are registered trademarks of BEA Systems, Inc. BEA Builder, BEA Campaign Manager for WebLogic, BEA eLink, BEA Liquid Data for WebLogic, BEA Manager, BEA WebLogic Commerce Server, BEA WebLogic Enterprise, BEA WebLogic Enterprise Platform, BEA WebLogic Enterprise Security, BEA WebLogic Express, BEA WebLogic Integration, BEA WebLogic JRockit, BEA WebLogic Personalization Server, BEA WebLogic Platform, BEA WebLogic Portal, BEA WebLogic Server Process Edition, BEA WebLogic Workshop and How Business Becomes E-Business are trademarks of BEA Systems, Inc.

All other trademarks are the property of their respective companies.

Contents

About This Document

Product Documentation on the dev2dev Web Site xii
Related Information xii
Contact Us! xii
Documentation Conventions xiv

1. Database Configuration Roadmap

Overview of Database Configuration for WebLogic Portal 1-1
Manually Creating Database Objects and JDBC Settings 1-2

2. Database Setup and Maintenance Overview

Character Sets and Sort Orders 2-2
Oracle 2-2
SQL Server 2-2
Sybase 2-3
DB2 2-3
Disk and Data Placement 2-3
Placement of Indexes 2-3
Data Storage for Tables with CLOB/BLOB/TEXT/IMAGE Data 2-4
Behavior Tracking 2-5
Reporting on Behavior Tracking data 2-5
Database Table Caching 2-5
Page and Block Size 2-6

Sybase Locking 2-6
Database Sizing 2-7
Behavior Tracking 2-7
Personalization 2-7
Portal Framework and WSRP 2-8
Content Management & Virtual Content Management (Versioning) 2-9
Content Search (Oracle) 2-10
Updating Database Statistics 2-10
Database Reorganizations 2-10
Database Backup and Recovery 2-10
Commerce Functionality in an XA Domain 2-11
WebLogic Portal Propagation Utility 2-11
Colocation of Database and WebLogic Portal 2-11

3. Using PointBase

PointBase Documentation 3-1
PointBase JAR Files 3-1
PointBase Tools 3-2
WebLogic Portal PointBase Databases 3-2
Administering the WebLogic Portal PointBase Database 3-2
Launching the PointBase Console from the Windows Start Menu 3-2
Launching the PointBase Console from the startPointBaseConsole Script 3-3

4. Using a Microsoft SQL Server Database

Configuring a Microsoft SQL Server Database 4-1
Manually Creating Database Objects 4-4
Manually Configuring Your Domain's JDBC Driver Settings 4-5
Creating a Separate Database for Behavior Tracking Events 4-7

5. Using an Oracle Database

Configuring an Oracle Database 5-2
Manually Creating Database Objects 5-5
Manually Configuring Your Domain's JDBC Driver Settings 5-6
Creating a Separate Database for Behavior Tracking Events 5-7
WebLogic Platform Support for Oracle9i RAC 5-8
Oracle 10g R2 Database Requires Manual Privilege Setup 5-8

6. Using a Sybase Database

Configuring a Sybase Database 6-1
Using a Supported Version 6-2
Defining an 8K Page Size 6-2
Running Upgrade Script for 7.0 to 8.1 6-2
Install and Configure the Sybase Client 6-2
Manually Creating Database Objects 6-6
Manually Configuring Your Domain's JDBC Driver Settings 6-7
Creating a Separate Database for Behavior Tracking Events 6-8

7. Using a DB2 Database

Configuring a DB2 Database 7-1
DB2 Configuration Parameter Minimum Settings 7-2
Manually Creating Database Objects 7-5
Manually Configuring Your Domain's JDBC Driver Settings 7-6
Creating a Separate Database for Behavior Tracking Events 7-7

8. Data Dictionary

Information Provided 8-1
Portal Database Components Covered 8-2
Behavior Tracking Database Objects 8-2

The BT_EVENT_TYPE Database Table 8-3
The BT_EVENT Database Table 8-4
The BT_EVENT_ACTION Database Table 8-9
Commerce Services Database Objects 8-9
Product Catalog Database Tables. 8-11
The CATALOG_ENTITY Database Table. 8-12
The CATALOG_PROPERTY_KEY Database Table. 8-12
The CATALOG_PROPERTY_VALUE Database Table 8-13
The WLCS_CATEGORY Database Table 8-14
The WLCS_PRODUCT Database Table 8-18
The WLCS_PRODUCT_CATEGORY Database Table. 8-22
The WLCS_PRODUCT_KEYWORD Database Table 8-22
Order and Discount Database Objects 8-24
The Order Processing Data Dictionary Tables. 8-26
The DISCOUNT Database Table 8-26
The DISCOUNT_ASSOCIATION Database Table 8-28
The ORDER_ADJUSTMENT Database Table 8-28
The ORDER_LINE_ADJUSTMENT Database Table. 8-29
The WLCS_CREDIT_CARD Database Table. 8-30
The WLCS_CUSTOMER Database Table. 8-32
The WLCS_ORDER Database Table. 8-34
The WLCS_ORDER_LINE Database Table 8-36
The WLCS_SAVED_ITEM_LIST Database Table 8-37
The WLCS_SECURITY Database Table. 8-37
The WLCS_SHIPPING_ADDRESS Database Table 8-38
The WLCS_SHIPPING_METHOD Database Table 8-39
The WLCS_TRANSACTION Database Table 8-40
The WLCS_TRANSACTION_ENTRY Database Table 8-42

Personalization Database Objects 8-42
The Portal Personalization Database Tables. 8-43
The GROUP_HIERARCHY Database Table 8-44
The GROUP_SECURITY Database Table 8-44
The USER_GROUP_CACHE Database Table 8-45
The USER_GROUP_HIERARCHY Database Table 8-45
The USER_PROFILE Database Table 8-46
The USER_SECURITY Database Table 8-46
The ENTITY Database Table 8-47
The PROPERTY_KEY Database Table 8-47
The PROPERTY_VALUE Database Table 8-48
The SEQUENCER Database Table. 8-49
The WEBLOGIC_IS_ALIVE Database Table 8-50
Data Synchronization Database Objects 8-50
The DATA_SYNC_APPLICATION Database Table 8-51
The DATA_SYNC_ITEM Database Table 8-52
The DATA_SYNC_SCHEMA_URI Database Table 8-53
The DATA_SYNC_VERSION Database Table 8-54
WebLogic Portal Services Database Objects 8-55
The Portal Services Database Tables 8-56
The AD_BUCKET Database Table 8-56
The AD_COUNT Database Table. 8-57
The PLACEHOLDER_PREVIEW Database Table 8-57
The MAIL_ADDRESS Database Table 8-58
The MAIL_BATCH Database Table. 8-58
The MAIL_BATCH_ENTRY Database Table 8-59
The MAIL_HEADER Database Table 8-59
The MAIL_MESSAGE Database Table 8-59

The SCENARIO_END_STATE Database Table 8-60
Portal Framework Database Objects 8-60
The Portal Framework Database Tables 8-62
The PF_BOOK_DEFINITION Database Table 8-63
The PF_BOOK_GROUP Database Table 8-65
The PF_BOOK_INSTANCE Database Table. 8-66
The PF_DESKTOP_DEFINITION Database Table. 8-67
The PF_DESKTOP_INSTANCE Database Table 8-68
The PF_LAYOUT_DEFINITION Database Table. 8-69
The PF_LOOK_AND_FEEL_DEFINITION Database Table 8-70
The PF_MARKUP_DEFINITION Database Table 8-71
The PF_MENU_DEFINITION Database Table. 8-72
The PF_PAGE_DEFINITION Database Table. 8-73
The PF_PAGE_INSTANCE Database Table 8-74
The PF_PLACEHOLDER_DEFINITION Database Table 8-75
The PF_PLACEMENT Database Table 8-76
The PF_PORTAL Database Table 8-77
The PF_PORTLET_CATEGORY Database Table. 8-77
The PF_PORTLET_CATEGORY_DEFINITION Database Table 8-78
The PF_PORTLET_DEFINITION Database Table 8-79
The PF_PORTLET_INSTANCE Database Table. 8-82
WSRP (Web Services for Remote Portlets) Objects 8-83
The PF_CONSUMER_PORTLETS Database Table 8-83
The PF_CONSUMER_PROPERTIES Database Table 8-84
The PF_CONSUMER_REGISTRY Database Table 8-84
The PF_PRODUCER_PROPERTIES Database Table. 8-86
The PF_PRODUCER_REGISTRY Database Table. 8-86
The PF_PROXY_PORTLET_INSTANCE Database Table. 8-88

The PF_PORTLET_PREFERENCE Database Table 8-89
The PF_PORTLET_PREFERENCE_VALUE Database Table 8-90
The PF_SHELL_DEFINITION Database Table. 8-91
The PF_THEME_DEFINITION Database Table 8-92
Content Management Database Objects 8-93
The Content Management Data Dictionary Tables 8-94
The CM_NODE Database Table. 8-94
The CM_OBJECT_CLASS Database Table. 8-95
The CM_PROPERTY Database Table 8-96
The CM_PROPERTY_CHOICE Database Table. 8-98
The CM_PROPERTY_DEFINITION Database Table 8-99
Content Management Virtual Database Objects 8-101
The CMV_NODE Table. 8-102
The CMV_NODE_ASSIGNED_ROLE Table. 8-103
The CMV_NODE_VERSION Table 8-104
The CMV_PROPERTY Table 8-104
The CMV_VALUE Table. 8-105
The CMV_NODE_VERSION_PROPERTY Table 8-107
Localization Database Objects 8-107
The Localization Dictionary Tables 8-108
The L10N_INTERSECTION Database Table 8-108
The L10N_LOCALE Database Table 8-109
The L10N_RESOURCE Database Table 8-110
The L10N_RESOURCE_TYPE Database Table 8-110
Tracked Anonymous User Database Objects 8-111
The Tracked Anonymous User Dictionary Tables 8-111
The P13N_ANONYMOUS_PROPERTY Database Table. 8-112
The P13N_ANONYMOUS_USER Database Table. 8-112

Entitlement Reference Database Objects 8-113
The Entitlement Reference Dictionary Tables 8-113
The P13N_ENTITLEMENT_APPLICATION Database Table 8-114
The P13N_ENTITLEMENT_POLICY Database Table 8-114
The P13N_ENTITLEMENT_RESOURCE Database Table 8-115
The P13N_ENTITLEMENT_ROLE Database Table 8-116
The P13N_DELEGATED_HIERARCHY Database Table 8-117

WebLogic Portal DDL Modules

WebLogic Portal DDL Modules A-1

Property Files and Database Scripts

The db_settings.properties file B-1
Definitions Section B-1
Database Parameters B-2
Scripts to Create or Upgrade Databases B-3

About This Document

This document explains how to set up and administer a database for WebLogic Portal. It covers the following topics:

- Chapter 1, "Database Configuration Roadmap," provides an introduction to database administration issues for WebLogic Portal.
- Chapter 2, "Database Setup and Maintenance Overview," describes special considerations that you should keep in mind as you set up your databases, and provides recommendations where appropriate.
- Chapter 3, "Using PointBase," provides information for setting up a SQL Server environment for WebLogic Portal, and instructions for switching from the PointBase database to SQL Server.
- Chapter 4, "Using a Microsoft SQL Server Database," provides information for setting up a SQL Server environment for WebLogic Portal, and instructions for switching from the PointBase database to SQL Server.
- Chapter 5, "Using an Oracle Database," provides information for setting up an Oracle environment for WebLogic Portal, and instructions for switching from the PointBase database to Oracle.
- Chapter 6, "Using a Sybase Database," provides information for setting up a Sybase environment for WebLogic Portal, and instructions for switching from the PointBase database to Sybase.

- Chapter 7, "Using a DB2 Database," provides information for setting up a DB2 environment for WebLogic Portal, and instructions for switching from the PointBase database to DB2.
- Chapter 8, "Data Dictionary," provides a complete list of all of the database schemas and dictionaries.
- Appendix A, "WebLogic Portal DDL Modules," describes the file naming convention used for the WebLogic Portal DDL files.
- Appendix B, "Property Files and Database Scripts," describes the database scripts that Portal includes and the processes they perform.

Product Documentation on the dev2dev Web Site

BEA product documentation, along with other information about BEA software, is available from the BEA dev2dev Web site:

`http://dev2dev.bea.com`

To view the documentation for a particular product, select that product from the list on the dev2dev page; the home page for the specified product is displayed. From the menu on the left side of the screen, select Documentation for the appropriate release. The home page displays the complete documentation set for the product and release that you select.

Related Information

Readers of this document may find the following documentation and resources especially useful:

- For general information about Java applications, go to the Sun Microsystems, Inc. Java Web site at `http://java.sun.com`.
- For general information about XML, go to the O'Reilly & Associates, Inc. `XML.com` Web site at `http://www.xml.com`.

Contact Us!

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

In your e-mail message, please indicate that you are using the documentation for BEA WebLogic Portal 8.1.

If you have any questions about this version of BEA WebLogic Portal, or if you have problems installing and running BEA WebLogic Portal, contact BEA Customer Support at http://support.bea.com. You can also contact Customer Support by using the contact information provided on the quick reference sheet titled "BEA Customer Support," 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
Ctrl+Tab	Indicates that you must press two or more keys simultaneously.
italics	Indicates emphasis or book titles.
monospace text	Indicates *user input*, as shown in the following examples: • Filenames: `config.xml` • Pathnames: *`BEA_HOME`*`/config/examples` • Commands: `java -Dbea.home=BEA_HOME` • Code: `public TextMsg createTextMsg(`
	Indicates *computer output*, such as error messages, as shown in the following example: `Exception occurred during event dispatching:java.lang.ArrayIndexOutOfBoundsException: No such child: 0`
monospace boldface text	Identifies significant words in code. *Example*: `void` **`commit`** `( )`
monospace italic text	Identifies variables in code. *Example*: `String` *`expr`*
{ }	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*: `java utils.MulticastTest -n` *`name`* `[-p` *`portnumber`*`]`
|	Separates mutually exclusive choices in a syntax line. The symbol itself should never be typed. *Example*: `java weblogic.deploy [list|deploy|update]`

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 "file1.cpp file2.cpp file3.cpp . . ."`
. . .	Indicates the omission of items from a code example or from a syntax line. The vertical ellipsis itself should never be typed.

CHAPTER 1

Database Configuration Roadmap

This document helps you set up and start using a database management system (DBMS) with WebLogic Portal.

By default, you can run the code samples provided with WebLogic Portal using the PointBase DBMS. PointBase is a pure-Java relational database management system that BEA includes with WebLogic Platform to allow you to run code samples. It is supported only for the design, development, and verification of applications; it is not supported for production server deployment. For more information about PointBase, see "Using PointBase" on page 3-1.

To maximize the performance of your database configuration, refer to the database section of the WebLogic Portal Performance Tuning Guide, located at

http://edocs.bea.com/wlp/docs81/perftune/index.html

Overview of Database Configuration for WebLogic Portal

Configuring a database for use with WebLogic Portal involves preparing the vendor database with several configuration scripts before connecting it to WebLogic Portal. Depending on your environment, the process may be entirely manual or you can use the WebLogic Configuration Wizard for part of the process.

To configure databases for use with WebLogic Portal, use the following steps.

1. Review related documentation to ensure that your configuration is supported.
   - http://edocs.bea.com/platform/suppconfigs/index.html
   - http://edocs.bea.com/platform/docs81/interm/relnotes.html

2. Create your vendor database(s). If you want to use behavior event tracking in a production environment, consider using a separate database for behavior event tracking.
3. Prepare the database for use with WebLogic Portal. BEA provides several sample initialization scripts that you must modify and run on the vendor database before using the database with WebLogic Portal.
4. After the database is configured, use the Configuration Wizard to create and load appropriate database objects and set JDBC driver settings at domain creation time. For more information about how to use the WebLogic Configuration Wizard, see http://edocs.bea.com/platform/docs81/confgwiz/index.html.

   You might want to perform this step manually under certain conditions; for details, see "Manually Creating Database Objects and JDBC Settings" on page 1-2.

Manually Creating Database Objects and JDBC Settings

In some cases, you may need to manually configure your database(s) without the use of the Configuration Wizard.

You can manually create database objects and JDBC settings in the following cases:

- When setting up your production database.
- If your desired database was not configured using the WebLogic Configuration Wizard.
- If after running the Configuration Wizard you decide to have your domain point to a different database.
- If you would like to refresh your database with the base configuration data that comes with the product.

  Note: BEA's database creation scripts first drop all database objects and then recreate them, which means all data added since your original installation will be lost. Upon completion of the database creation scripts, only the base configuration data that is needed for the product will exist.
- When you want to create only a subset of Portal database objects, for example to create only Behavior Tracking database objects for a particular database.

CHAPTER 2

Database Setup and Maintenance Overview

This section identifies considerations that the database administrator may want to be aware of in setting up and maintaining WebLogic Portal databases.

For a discussion of database environment considerations for team development, see the section "Creating a Shared Portal Domain" in the Production Operations Guide at http://e-docs.bea.com/wlp/docs81/prodOps/index.html.

For additional details on database-specific considerations and recommendations, see the individual database sections in this document.

Information included in this section includes:

- Character Sets and Sort Orders
- Disk and Data Placement
- Database Sizing
- Updating Database Statistics
- Database Reorganizations
- Database Backup and Recovery
- Commerce Functionality in an XA Domain
- WebLogic Portal Propagation Utility
- Colocation of Database and WebLogic Portal

Character Sets and Sort Orders

A database character set determines which languages can be represented in the database. A database's sort order, or collation, determines the rules by which characters are sorted and compared.

For a globalized WebLogic Portal application, you should plan a database's character set and/or sort order before you set up the database. Changing an existing database's character set and sort order can be a time- and resource-intensive task. Typically a change can be made only if the target character set is a subset of the source or original.

The following sections describe character set and sort order considerations for Portal-supported databases.

Oracle

For Oracle, you define the character set when you create the database.

- For Oracle 9i and 10g databases against which a globalized WebLogic Portal application will run, a commonly used character set is `AL32UTF8`.
- For Oracle 8.1.7 databases, the `UTF8` character set is a common selection.

You can retrieve information about an Oracle database's character set and sort order by querying the `SYS.NLS_DATABASE_PARAMETERS` table.

SQL Server

For SQL Server, you can define both an instance level and a database level character set and sort order. The default instance level character set for a SQL Server is determined by the locale setting for the Windows operating system. The following sample output from the `sp_helpsort` stored procedure shows a common instance level sort order:

Latin1-General, case-insensitive, accent-sensitive, kanatype-insensitive, width-insensitive for Unicode Data, SQL Server Sort Order 52 on Code Page 1252 for non-Unicode Data.

The sample database creation script is located at *`WL_HOME`*`\portal\db\sql_server\2000\admin\create_database.sql` defines the WebLogic Portal database with the settings `COLLATE SQL_Latin1_General_CP1_CS_AS`. Note that this specifies a case-sensitive definition. Setting the database to case-sensitive allows Content Management queries to function the same for SQL Server as they do for other databases. If Content Management search queries are not written for a case-sensitive environment, you may obtain unexpected results.

You can use the `sp_helpdb <dbname>` stored procedure to display a SQL Server database's character set and sort order.

Sybase

For Sybase, you define the character set and sort order when you create the Sybase instance. A commonly used Sybase character set and sort order definition is:

Character Set = 2, cp850 Code Page 850 (Multilingual)

Sort Order = 50, bin_cp850 Binary ordering, for use with Code Page 850 (cp850).

You can use the `sp_helpsort` stored procedure to display information about an instance's character set and sort order.

DB2

For DB2 databases, you can set character encoding for each database individually. The default character encoding is determined by the operating system.

A commonly used DB2 character encoding is `UTF-8`.

You can examine the `DB2 CODESET` configuration parameter to determine a database's codeset.

Disk and Data Placement

The following sections describe special considerations, recommendations, and requirements for index placement, CLOB/BLOB/TEXT/IMAGE data storage, and behavior tracking.

Placement of Indexes

For Oracle databases, use the `rebuild_indexes.sql` script to place indexes into a WEBLOGIC_INDEX tablespace. Run this script manually after creating the WebLogic Portal database to move indexes into their own tablespace.

For SQL Server and Sybase, as of 8.1 SP4 the Data Definition Language (DDL) for non-clustered indexes is defined with `ON WEBLOGIC_INDEX` to place indexes into their own file group or segment. You must define the `WEBLOGIC_INDEX` file group or segment before running WebLogic Portal database creation or upgrade scripts. For detailed information about SP4 database changes, see the Upgrade Guide at http://e-docs.bea.com/wlp/docs81/upgrade/index.html.

Data Storage for Tables with CLOB/BLOB/TEXT/IMAGE Data

This section describes recommendations and special considerations for tables that contain CLOB/BLOB (Oracle/DB2) or TEXT/IMAGE (Sybase/SQL Server) data.

As a general rule, for optimal performance with any of the supported Portal database platforms, CLOB/BLOB/TEXT/IMAGE table data should be physically stored separately from non-CLOB/BLOB/TEXT/IMAGE data. However, to simplify deployment and to allow for flexibility in your configuration, the default Portal database schema is not deployed with separated data. To determine if your specific application is a candidate for separating CLOB/BLOB/TEXT/IMAGE storage, assess the following considerations in your environment:

- Content management usage - If you are using the out-of-the-box content management system heavily and you are storing and retrieving many large documents, you may want to modify your storage allocations for the tables that have the CM and CMV prefixes.

  The corresponding tables and columns for this change are:

  - `CMV_VALUE.BLOB_VALUE`
  - `CM_PROPERTY.BLOB_VALUE`
  - `CM_PROPERTY_CHOICE.BLOB_VALUE`

- Behavior tracking - If you have behavior tracking turned on, you may want to modify storage allocations for behavior tracking tables.

  The corresponding table and column for this change is `BT_EVENT.XML_DEFINITION`.

- Physical database storage - If your environment has a separate disk controller and drive that can be dedicated to CLOB/BLOB/TEXT/IMAGE storage, you might see greater performance impacts than if you share a controller.

- RAID storage - If you are using RAID, depending on the configuration you may see minimal if any improvement by changing storage allocations. Follow your database vendor recommendations based on your specific configuration.

If you want to change your storage allocations, see your database documentation for specific details on changing CLOB/BLOB/TEXT/IMAGE settings.

The following additional tables contain CLOB/BLOB or TEXT/IMAGE data types; you can experiment with storage changes to determine the effect on performance.

```
AD_BUCKET.AD_QUERY
CATALOG_PROPERTY_VALUE.BLOB_VALUE
DATA_SYNC_ITEM.XML_DEFINITION
```

```
DISCOUNT.DISCOUNT_RULE
MAIL_MESSAGE.MESSAGE_TEXT
P13N_ANONYMOUS_PROPERTY.PROPERTY_VALUE
P13N_DELEGATED_HIERARCHY.ENTITLEMENT_DOCUMENT
PF_CONSUMER_REGISTRY.REGISTRATION_STATE
PF_PROXY_PORTLET_INSTANCE.PORTLET_STATE
PLACEHOLDER_PREVIEW.XML_DEFINITION
PROPERTY_VALUE.BLOB_VALUE
```

Behavior Tracking

Behavior tracking is typically used to track visitor behavior by recording events.

Due to the large number of rows that can be written to the BT_EVENT table when behavior tracking is enabled, you might want to use a separate database (or tablespace and schema, depending upon your DBMS) to store behavior tracking data.

For each database type other than PointBase, the following sections include instructions for creating a separate database for behavior tracking events:

- "Using a Microsoft SQL Server Database" on page 4-1
- "Using an Oracle Database" on page 5-1
- "Using a Sybase Database" on page 6-1
- "Using a DB2 Database" on page 7-1

Reporting on Behavior Tracking data

Some third-party behavior tracking reporting tools extract data from the BT_EVENT table data and place it into another set of database tables. For more information on reporting and analytics tools, see the Portal Solutions Catalog at http://dev2dev.bea.com/products/wlportal/psc/Reporting_Analytics_BI.jsp.

Database Table Caching

Some databases provide the ability to cache or "pin" database tables into memory. Choose a database caching implementation based on the WebLogic Portal components that are deployed for your application and how you deploy them.

The following text summarizes the support provided by each database type for caching or pinning database tables. Depending on your environment, focus your caching strategy on small tables and frequently referenced tables.

- Oracle's touch-count algorithm makes it unnecessary to pin specific tables to buffers or database caches. You can adjust Oracle instance parameters to influence buffer behavior.
- SQL Server provides `DBCC PINTABLE (database_id, table_id)` to pin tables to the buffer cache.
- Sybase provides the ability to define multiple database cache and to selectively bind tables, indexes and logs to those caches. Use `sp_objects_stats` to identify hot objects that may benefit from their own cache. Use a Sybase monitoring tool or `sp_sysmon` to determine if cache hit ratios are acceptable.
- DB2 uses `BUFFERPOOLs` to cache database data and describes configuring bufferpools as the single most important tuning area. DB2 allows you to assign individual tables and indexes to a `BUFFERPOOL`.

Page and Block Size

Note: See the individual database chapters for additional details on database-specific considerations.

For Oracle and SQL Server databases, an 8K page/block size is the default.

For DB2 an 8K bufferpool is defined for the WebLogic Portal tables and indexes that require this larger pool size (higher than the 4K default).

For Sybase databases an 8K pagesize is required for several WebLogic Portal tables and indexes.

Sybase Locking

A Sybase instance's default locking mechanism is "ALL PAGES". For concurrency, locking can also be defined for each table in a Sybase database.

In 8.1 SP4 the following WebLogic tables are defined with `LOCK DATAROWS` for Sybase. Other WebLogic Portal tables for Sybase (excluding those defined in `collaboration_create_tables.sql`) are defined with `LOCK DATAPAGES`.

- CATALOG_ENTITY
- ENTITY

- SEQUENCER
- L10N_INTERSECTION
- PF_DESKTOP_INSTANCE
- PF_PLACEMENT

Based on your usage of WebLogic Portal components with a Sybase database, you may decide to modify additional tables to `LOCK DATAROWS`.

Database Sizing

The size required for your WebLogic Portal database(s) depends on many factors, including the following:

- The components of WebLogic Portal that are deployed for your application and the method you use to deploy them. By default, all WebLogic Portal database objects are created for each domain. If a component of WebLogic Portal is not deployed, its database objects will exist, but will not contain any data rows.
- The number of WebLogic Portal application users in your environment.
- The degree to which customization or personalization is allowed for your end users' WebLogic Portal resources. For example, you might ask these questions about user personalization:
  - Can end users create their own collaboration portlets such as My Mail Portlet, My Task List Portlet, and so on?
  - Can end users create their own portal resource views?

  Note: Portal resource view data is stored in the Portal Framework tables named `PF_<resource_type>_INSTANCE`.

The following sections describe some WebLogic Portal database tables that you should monitor closely for growth due to increasing data volumes.

Behavior Tracking

The BT_EVENT table can grow significantly if Behavior Tracking is enabled.

Personalization

The following database tables store personalization data; monitor them for data growth.

- ENTITY — This table will grow as you define new users and groups for your WebLogic Portal application.
- PROPERTY_VALUE — This table will grow as property values are added to a user profile. This table can become quite large when a large number of users exist with a large number of property values per user. For a discussion of user and group profile values, see the Administration Portal online help at http://e-docs.bea.com/wlp/docs81/adminportal/help/UG_UserProfile_Edit.html.

Portal Framework and WSRP

The following tables involve the Portal Framework and/or Web Services for Remote Portlets (WSRP) and can potentially be high volume tables; monitor them for data growth.

- PF_BOOK_INSTANCE — This table identifies instances of the `BOOK_DEFINITION`. There is always at least one book instance, namely the primary instance; all other instances represent customization by administrators or end users. This table can grow significantly if extensive user customization occurs.
- PF_BOOK_GROUP — This table represents child pages or book placements on the parent book. A single record in the table represents one placement on a book. This table can grow significantly if extensive user customization occurs.
- PF_PLACEMENT — This table tracks the portlets on a specific portal page for each desktop. Any time a visitor modifies the position of portlets on a page, a row will be inserted for each portlet that exists on the portal page for the user's custom desktop. This table can grow significantly because of portal customization; this growth can be expressed using the equation *number of users * number of pages * number of portlets = number of rows*.
- PF_PORTLET_PREFERENCE_VALUE — This table identifies preference values for the portlet instance. This table can grow significantly if extensive user customization occurs.
- PF_PORTLET_INSTANCE — A portlet definition has at least one portlet instance that is known as the primary instance. Every time a portlet is dragged onto a page, a new instance of the portlet is being created and a column is inserted into the table. If, for example, an administrator drags a portlet onto a page and then a user modifies the portlet by setting the default to "minimized," then another instance is created and a column inserted. (Subsequent changes to the portlet instance do not create a new instance/row.) This growth can be expressed using the equation *number of portlet definitions * number of instances = number of rows*.

- PF_DESKTOP_INSTANCE — This table identifies a customized or localized instance of a desktop. This table can grow significantly; a row is added for each entitled desktop, and possibly a row for each unique visitor who customizes the site. This growth can be expressed using the equation *number of portals * number of desktops + number of visitors who customize = number of rows*.

Content Management & Virtual Content Management (Versioning)

The following database tables store content management (CM) and versioning (CMV) data; monitor them for data growth.

- CM_PROPERTY_DEFINITION — This table defines the shape of a property; it describes the property type (BLOB, Boolean, Varchar, Float, Date, Number), whether it is required, whether it is editable, the default value, and restricted values, if applicable. This table can grow significantly if content management is used to store large quantities of data.
- CMV_NODE — This table uniquely identifies a content-managed node from a BEA repository (from the CM_NODE table) that has been versioned and is being edited within the Content Management Virtual Repository. This table can grow significantly if content management is used to store large quantities of data.
- CMV_NODE_VERSION — This table uniquely identifies all the versions of a mode within the Content Management Virtual Repository. This table can grow significantly if Content Management versioning is enabled.
- CMV_NODE_VERSION_PROPERTY — This table uniquely identifies a relationship between a CMV_NODE_VERSION and CMV_PROPERTY. This table will likely have the largest number of rows within Content Management if versioning is enabled. It may not be the largest table because it is a cross-reference table.
- CMV_PROPERTY — This table uniquely identifies a property that can be associated with a node version. For example, some properties of a book might be author, title, and subject. This table can grow significantly if Content Management versioning is enabled.
- CMV_VALUE — This table uniquely identifies a value for a given property. For example, a property `SUBJECT` for a `BOOK` might have a value of `FINANCE`. This table can grow significantly if Content Management is used to store large quantities of data. This table will likely be the largest Content Management table because it stores the actual content associated with each Content Node.

Content Search (Oracle)

Content search within WebLogic Portal typically performs best when accessing database indexes associated with the content repository. In some situations the Oracle optimizer, based on database statistics, will choose to perform tablespace scans instead of using indexes to access the data. Most often this will result in a much slower response time than if an index was used.

To improve response time, verify that the following Oracle initialization parameter reads:

```
optimizer_mode=choose.
```

In addition to ensure that Oracle indexes are given greater preference over tablespace scans, the following two initialization parameters can be altered:

```
optimizer_index_cost_adj (range 0-100)
optimizer_index_caching (range 0-100)
```

Refer to your Oracle documentation for the impact of each of these settings to make sure it is right for your installation.

Updating Database Statistics

Each DBMS has its own utility or commands for updating the database statistics used by its query optimizer. A DBA should schedule periodic jobs to maintain database statistics.

Database Reorganizations

The statistics that you collect for your database often provide information on the organization of tables and indexes in your database.

Normal DML operations (for example, `DELETE`, `INSERT`, `UPDATE`) that the WebLogic Portal application performs can affect table and index organization; this can affect database performance. Refer to your database vendor product documentation for details on utilities that are available for table and index reorganization and for information on determining when a re-organization should occur.

Database Backup and Recovery

Use the same procedures for backup and recovery of WebLogic Portal databases as you use for other data. The following text lists some general recommendations:

- Store your E-Business Control Center data and J2EE resources in source control, and back up the source control database.

- Back up your WebLogic Portal database according to your DBMS vendor's recommendations.

Perform periodic test restores to ensure that your backups are sound.

Commerce Functionality in an XA Domain

If you are using the optional commerce functionality in a Portal domain configured for XA, then you must move the weblogic.jdbc.jts.commercePool JNDI name from the portalFrameworkPool to the cgDataSource-nonXA JDBC Tx Data Source. For information about using commerce functionality, see "Adding Commerce Services to an Application" in the WebLogic Workshop Help.

WebLogic Portal Propagation Utility

The Propagation Utility guides you through the process of propagating the configuration contents, including portal framework, datasync, and security data, of one portal domain environment to another. For example, the Propagation Utility can play a role whenever you move a portal application from a staging environment to the production environment.

You should be aware of the usage of this utility, and you may need to back up databases prior to propagation. For more information about this utility, contact BEA customer support.

Colocation of Database and WebLogic Portal

To achieve the best possible performance, your database and your WebLogic Portal installation should be colocated and connected using a high speed network connection to avoid network latency issues. This recommendation is especially important when you use the Propagation Utility, to ensure that a large propagation process can complete successfully.

CHAPTER 3

Using PointBase

PointBase is the default database that BEA provides. It is used for the BEA sample domains, and it is the default database used when you create a domain with the Configuration Wizard. WebLogic Portal does not support PointBase for production deployments.

PointBase runs on its own server that must be running for your applications to access it. When you start WebLogic Portal server to run your applications, the PointBase server starts automatically.

PointBase Documentation

PointBase documentation, for the version of PointBase currently supported by WebLogic Server, is distributed with WebLogic Server in PDF form in the *WL_HOME*\common\eval\pointbase\docs directory. The PointBase documentation consists of the following three manuals:

- PointBase Console Guide
- PointBase System Guide
- PointBase Developer's Guide

PointBase JAR Files

Refer to the section titled "PointBase JAR Files" in the PointBase System Guide for information on the JAR files provided with WebLogic Server in the *WL_HOME*\common\eval\pointbase\lib directory.

PointBase Tools

Scripts for starting the PointBase server and the PointBase console are distributed with WebLogic Server in the *WL_HOME*\common\eval\pointbase\tools directory. Scripts are called by start scripts in the sample domains and start scripts contained in any domain created by the Configuration Wizard. These PointBase start scripts simplify starting the PointBase Server and Console within WebLogic domains.

WebLogic Portal PointBase Databases

PointBase stores all data in .dbn files and all log information in .wal files. Database properties are stored in PointBase.ini files. Data files for WebLogic Portal are named **workshop.dbn** and log files for WebLogic Portal are named **workshop$1.wal**. Pre-built PointBase data, log, and PointBase.ini files for WebLogic Portal samples are included in the following directory:

WL_HOME\samples\domains\portal

By default domains created using the Configuration Wizard with the Basic WebLogic Portal Domain template would create PointBase data and log files in the following directory:

BEA_HOME\user_projects\portalDomain

Administering the WebLogic Portal PointBase Database

You can administer PointBase using the PointBase administrative console, or any third-party database visualization and management tool that can connect using JDBC.

You can launch the PointBase Console either from the Windows Start Menu or by executing the startPointBaseConsole.cmd/.sh script located in the domain directory.

Prior to launching the PointBase Console, ensure that WebLogic Server for the domain is running. You cannot use the PointBase Console unless WebLogic Server is running.

Launching the PointBase Console from the Windows Start Menu

Go to Start →Programs →BEA WebLogic Platform 8.1 →Examples →WebLogic Portal → PointBase Console.

To launch the PointBase Console for the Portal Examples, or, if you added Start menu options for a domain created by the Configuration Wizard, navigate to that domain's PointBase Console menu option.

Launching the PointBase Console from the startPointBaseConsole Script

1. Change directories to *WL_HOME*\samples\domains\portal.
2. Execute the appropriate start script — startPointBaseConsole.cmd or startPointBaseConsole.sh — to launch the PointBase Console for the Portal Examples. For a domain created by the Configuration Wizard navigate to that domain's home directory and execute the startPointBaseConsole.cmd/.sh script.
3. When the PointBase Console starts, it prompts you to enter connection parameters to properly connect to the database. Enter the following connection information; you also need this information if you use a third-party product to access the PointBase database:
   - **Driver**: com.pointbase.jdbc.jdbcUniversalDriver
   - **URL**: jdbc:pointbase:server://localhost:9093/workshop
   - **User**: weblogic
   - **Password**: weblogic

CHAPTER 4

Using a Microsoft SQL Server Database

This section describes the steps necessary to use a Microsoft SQL Server database with WebLogic Portal 8.1, and includes information on the following subjects:

- Configuring a Microsoft SQL Server Database
- Manually Creating Database Objects
- Manually Configuring Your Domain's JDBC Driver Settings
- Creating a Separate Database for Behavior Tracking Events

Review this entire chapter and any release notes before proceeding. The steps in this chapter should be performed by a database administrator.

Note: For additional database setup information, see "Managing WebLogic Platform Database Resources" at http://e-docs.bea.com/platform/docs81/db_mgmt/db_resource_mgmt.html.

Configuring a Microsoft SQL Server Database

Before proceeding, be sure you have read "Overview of Database Configuration for WebLogic Portal" on page 1-1.

To configure a SQL Server database:

1. *This step is required only if you are not planning to use the Configuration Wizard to create the database objects for a new domain*.

   Install the SQL Server client on the WebLogic Platform host and do the following:

a. Configure it for access to your SQL Server database.

b. Ensure that you can connect to your SQL Server database using the OSQL utility.

   See your SQL Server documentation for details.

2. Verify that security authentication settings for the SQL Server are set to "SQL Server and Windows."

   a. From Enterprise Manager, right-click the desired SQL Server.

   b. Select Properties, then select the Security tab.

   c. Under authentication, ensure that SQL Server and Windows is selected.

3. Prepare the SQL Server database. The database creation scripts install domain-specific tables. It is recommended that you work with a SQL Server system or database administrator to adjust the sample scripts and create database devices, file groups, databases, and database users for your SQL Server environment.

Notes: Multiple databases are required if you have multiple domains, or to run multiple environments using the same SQL Server instance (for example, if you want to run development and system test from a single SQL Server installation).

Be sure to back up your database(s) before installing any new database objects. See your database documentation for details.

a. Review and modify the provided sample scripts to suit your environment. These scripts are available in the *WL_HOME*\portal\db\sql_server\2000\admin directory.

The following table describes the script names and the usage notes for each script.

Script Name	Description
create_database.sql	Creates the `WEBLOGIC` database, the `WEBLOGIC_INDEX` file group, and `WEBLOGIC` database owner (dbo) user login. An alias is created to make `WEBLOGIC` the dbo user in the database. Sets the WebLogic database as the default database for the WebLogic user. **Usage Notes:** Edit the script to change database names, database owner user, and password. The default names are the following: • database name: WEBLOGIC • database owner user: WEBLOGIC • password: WEBLOGIC You also need to edit the script to reflect valid disk locations for `DATA` devices, `LOG` devices, and the `WEBLOGIC_INDEX` file group; you may also need to adjust file sizes. Put `DATA` and `LOG` files on separate physical disks and away from any system database files, unless you are using RAID devices.
statistics.sql	Runs `sp_updatestats` to compute database statistics needed for the database optimizer. You should update database statistics periodically. (This is done by default for SQL Server databases with the AUTO_UPDATE_STATISTICS database option.) When set to ON (default), existing statistics are automatically updated when the data in the tables has changed. When set to OFF, existing statistics are not automatically updated. You must manually update statistics. The AUTO_UPDATE_STATISTICS option setting is stored in the IsAutoUpdateStatistics property of the DATABASEPROPERTYEX function.

install_report.sql	Builds an informational installation report about the database objects created in the WEBLOGIC schema.
bt_create_database.sql	Create the WEBLOGIC_EVENT database and WEBLOGIC_EVENT database owner user login. An alias is created to make WEBLOGIC_EVENT the database owner (dbo) user in the database. **Usage Notes:** Edit the script to change database names, database owner user, and password. The default names are the following: • database name: WEBLOGIC_EVENT • database owner user: WEBLOGIC_EVENT • password: WEBLOGIC_EVENT You also need to edit the script to reflect valid disk locations for the DATA and LOG devices, or to adjust file sizes. Put DATA and LOG files on separate physical disks and away from any system database files.

b. Run `create_database.sql` using `OSQL` as a user with System Administrator privileges (that is, the sa user). For example:

```
osql –Usa -SSQLSERVER -e -icreate_database.sql -ocreate_database.log
```

The output from running `create_database.sql` is written to `create_database.log`. Verify that there are no errors in the log file before proceeding.

Manually Creating Database Objects

You can either manually create database objects or use the Configuration Wizard. For details, see "Overview of Database Configuration for WebLogic Portal" on page 1-1.

Note: If you choose to use the WebLogic Configuration Wizard to configure and connect to the database that you will use to support WebLogic Portal, see http://edocs.bea.com/platform/docs81/confgwiz/index.html.

The scripts to create Microsoft SQL Server database objects were designed to run in a Windows environment (they use the OSQL utility to create Microsoft SQL Server database objects). If you are using UNIX version of WebLogic Server with a Microsoft SQL Server database and do not have WebLogic products also installed on Windows, contact BEA support for assistance.

To manually create WebLogic Portal database objects, use the following steps:

1. Verify that you can connect to the target database server with a valid user ID and password. For example:

   ```
   osql -SSQLSERVER -Uuserid -Ppassword
   ```

2. Open your domain's `db_settings.properties` file for edit and comment out the database settings for PointBase.
3. Uncomment the database settings for SQL Server and update the following settings for your database:
   - `server=`
   - `dblogin=`
   - `password=`
4. Initialize the database with the new settings.
   a. For Windows, navigate to the *`BEA_HOME`*`\user_projects\domains\portalDomain` directory and double-click the `create_db.cmd` file.
   b. Verify the results in the `create_db.log` file.

Note: If you are using the sample domain, run the create_db.cmd file from the following directory: *`WL_HOME`*`\samples\domains\portal`.

5. Follow the steps in "Manually Configuring Your Domain's JDBC Driver Settings" on page 4-5.

Manually Configuring Your Domain's JDBC Driver Settings

You can either manually configure your domains JDBC driver settings using the WebLogic Server Console, or use the Configuration Wizard. For more information, see "Overview of Database Configuration for WebLogic Portal" on page 1-1.

To manually configure your JDBC driver settings using WebLogic Server Console:

1. Start the WebLogic Server for your domain.
2. Login to the WebLogic Server Console.
3. Configure your new connection pools.
   a. Go to Services →JDBC →Connection pools.
   b. Click Configure a new connection pool.

c. Select the appropriate database type and non-XA database driver from the drop-down list boxes and click Continue. For more information, see the Supported Configuration documentation for JDBC drivers supported by WebLogic Portal, http://edocs.bea.com/platform/docs81/support/supp_plat.html#1085671.

   For an XA configuration, see "Creating XA Domains Using Configuration Templates" in the "Creating WebLogic Configurations Using the Configuration Wizard" documentation, http://edocs.bea.com/platform/docs81/confgwiz/index.html.

d. Choose a name for the new connection pool (For example: cgPoolN) and fill in the blanks for your vendor database. Click Continue.

e. Test your connection to verify that you can successfully connect to your database.

f. Create and deploy your new connection pool.

   Note: You must maintain a one-to-one mapping of JDBCTxDataSource to JDBC connection pool in the domain's `config.xml` file. Create one new JDBC connection pool for each JDBCTxDataSource and another JDBC connection pool for the domain's JDBCDataSources.

4. Update your data sources.

   a. From Services →JDBC →Data Sources, click each data source and switch each to the newly created connection pool. Make sure that you apply each change.

   b. Verify that each data source is changed by clicking on Data Sources and then verifying that Pool Name has been set to the new connection pool for each.

5. From Services →JMS →Stores →cgJMSStore, switch cgJMSStore to use the new connection pool.

6. Stop your domain's WebLogic Server, then restart it.

7. In the WebLogic Server Console, delete the original connection pools.

   a. Go to Services →JDBC →Connection Pools.

   b. Right-click each connection pool and select Delete.

Creating a Separate Database for Behavior Tracking Events

For improved performance, you might want to store behavior tracking events in a different location from other WebLogic Portal database objects. For more information about behavior tracking, see http://e-docs.bea.com/wlp/docs81/adminportal/help/SA_BehavTrackServ.html.

Note: By default, behavior tracking database objects are created in the same database as other WebLogic Portal database objects. You need to perform these steps only if you are configuring a separate database for behavior tracking events.

1. Edit the `bt_create_database.sql` file for your environment, as indicated in the instructions contained in the file.
2. Run `bt_create_database.sql` using `OSQL` as a user with system administrator privileges. For example:

   ```
   osql -Usa -SSQLSERVER -e -ibt_create_database.sql
   -obt_create_database.log
   ```

   The output from running `bt_create_database.sql` is written to `bt_create_database.log`. Verify that there are no errors in the log file before proceeding.
3. Navigate to the appropriate database directory based on your environment: *`WL_HOME`*`\portal\db\sql_server\2000`
4. Connect as the user `WEBLOGIC_EVENT` and run the following scripts:
   - `bt_create_tables.sql`
   - `bt_create_fkeys.sql`
   - `bt_create_indexes.sql`
   - `bt_create_views.sql`
   - `bt_create_triggers.sql`
5. Run the following script from the path *`WL_HOME`*`\portal\db\data\required`:
   - `bt_insert_system_data.sql`
6. Configure a connection pool to access your behavior tracking database and associate the p13n_tracking JDBC data source with that connection pool. Follow the steps in "Manually Configuring Your Domain's JDBC Driver Settings" on page 4-5.

CHAPTER 5

Using an Oracle Database

This section describes the steps necessary to use an Oracle database with WebLogic Portal 8.1, and includes information on the following subjects:

- Configuring an Oracle Database
- Manually Creating Database Objects
- Manually Configuring Your Domain's JDBC Driver Settings
- Creating a Separate Database for Behavior Tracking Events
- WebLogic Platform Support for Oracle9i RAC
- Oracle 10g R2 Database Requires Manual Privilege Setup

Review this entire chapter and any release notes before proceeding. Typically, the steps described in this chapter should be performed by an Oracle system administrator or a database administrator.

Notes:

- For additional database setup information, see "Managing WebLogic Platform Database Resources" at http://e-docs.bea.com/platform/docs81/db_mgmt/db_resource_mgmt.html.
- For performance tuning information for Oracle databases, see "Oracle Tuning Tips" at http://e-docs.bea.com/wlp/docs81/perftune/2ptgeneral.html

Configuring an Oracle Database

Before proceeding, be sure that you have read "Overview of Database Configuration for WebLogic Portal" on page 1-1.

Note the following considerations when defining your Oracle instance and databases:

- Be sure that you are using a supported version; for details, see http://edocs.bea.com/platform/docs81/support/supp_plat.html.
- Define a blocksize of at least 8K for increased performance.
- Oracle configuration settings can impact Content Search performance; for more information, see the Performance Tuning Guide at

  http://e-docs.bea.com/wlp/docs81/perftune/2ptgeneral.html.

1. Install the Oracle client software on the WebLogic Platform host.

Note: If you plan to use the Configuration Wizard to create the database objects for a new domain, you do not need to install the Oracle Client.

   a. Configure a Local Net Service to access the target Oracle instance.

   b. Be sure that Oracle environment variables are defined, and that the Oracle bin directory is included in the `$PATH` variable.

   c. Verify that you can connect to the target Oracle database schema using SQLPlus.

2. Prepare the Oracle database and schema. The database creation scripts install domain-specific tables for each. It is recommended that you work with a database administrator to adjust the `SAMPLE` scripts, and to create the database schema owner users and tablespaces needed for your environment.

Notes: Multiple database schemas are required if you have multiple domains, or to run multiple environments using the same Oracle instance (for example, if you want to run development and system test from a single Oracle installation).

Be sure to back up your database before installing any new database objects. See your database documentation for details.

   a. Edit the sample scripts provided in: *`WL_HOME`*`/portal/db/oracle/817/admin` to suit your environment.

The database creation scripts install domain-specific tables for each. It is recommended that you work with a database administrator to adjust the sample scripts, and to create the database schema owner users and tablespaces needed for your environment.

b. Review the Description and Usage Notes for each script.

Script Name	Description

create_tablespaces.sql	Creates data and index tablespaces.
	Usage Notes: Edits are required to modify the pathnames for the DATA_PATHNAME and INDEX_PATHNAME variables to match your local directory path structures. For example, on a UNIX system, if two disks are mounted as /usr1 and /usr2 and the Oracle SID is PROD, use the following pathnames:
	`DEFINE DATA_PATHNAME=/usr1/oradata/PROD`
	`DEFINE INDEX_PATHNAME=/usr2/oradata/PROD`
	Edits are also required if you want to change the tablespace names. The following defaults are used:
	• `WEBLOGIC_DATA`: tables for WebLogic Portal and/or WebLogic Platform
	• `WEBLOGIC_INDEX`: indexes for WebLogic Portal and/or WebLogic Platform

create_users.sql	Creates a `WEBLOGIC` schema owner user, establishes the users password, default and temporary tablespaces and grants privileges to that user. **Usage Notes:** Edits are required to change the schema owner user name, password and tablespace names. The following defaults are used: • database user = WEBLOGIC • database password = WEBLOIGIC • default tablespace = WEBLOGIC_DATA • temporary tablespace = TEMP
rebuild_indexes.sql	Rebuilds `WEBLOGIC` (schema user) indexes to move them from the `WEBLOGIC_DATA` tablespace to the `WEBLOGIC_INDEX` tablespace.
statistics.sql	Runs `analyze_schema` to compute database statistics needed for the Oracle optimizer. Run `analyze_schema` whenever any significant changes in database data occur. Your database administrator typically schedules `analyze_schema` to run periodically in your environment.
install_report.sql	Builds an informational installation report about the database objects created in the schema.
db_size.sql	Builds a report showing free space in database tablespaces.

Script Name	Description
bt_create_tablespaces.sql	Creates the tablespace for behavior event tracking. **Usage Notes:** Edits are required to modify the pathnames for the EVT_DATA_PATHNAME and INDEX_PATHNAME variables to match your local directory path structures. • `WEBLOGIC_DATA`: tables for WebLogic Portal and/or WebLogic Platform • `WEBLOGIC_INDEX`: indexes for WebLogic Portal and/or WebLogic Platform
bt_create_users.sql	Creates a behavior event tracking user; establishes the user's password, and default and temporary tablespaces; and grants privileges to that user. **Usage Notes:** Edits are required to change the schema owner user name, password and tablespace names. Edits are required to change file sizes and device names. The following defaults are used: • database user: WEBLOGIC_EVENT • password: WEBLOGIC_EVENT

c. To run these scripts from a shell, change directories to:

```
WL_HOME/portal/db/oracle/817/admin
```

d. Start SQL*Plus as the system user. For example:

```
sqlplus system/manager@MYDB
```

e. From SQL*Plus, execute the create_tablespaces.sql script. using the @ sign. For example:

```
@create_tablespaces.sql
```

f. From SQL*Plus, execute the create_users.sql script using the @ sign. For example,

```
@create_users.sql
```

3. Follow the steps in "Manually Creating Database Objects" on page 5-5.

Manually Creating Database Objects

You can either manually create database objects or use the Configuration Wizard; for more information, see "Overview of Database Configuration for WebLogic Portal" on page 1-1.

Note: If you choose to use the WebLogic Configuration Wizard to configure and connect to the database that you will use to support WebLogic Portal, see http://edocs.bea.com/platform/docs81/confgwiz/index.html.

To manually create WebLogic Portal database objects, use the following steps:

1. Use the following command to verify that you can connect to the target database server with a valid user ID and password:

   ```
   sqlplus user_ID/password@DB_SID
   ```

2. Open your domain's db_settings.properties file for edit, and comment out the database settings for PointBase.
3. Uncomment the database settings for Oracle and update the following settings for your database:
   - `server=`
   - `dblogin=`
   - `password=`
4. Initialize the database with the new settings.
   a. For Windows, navigate to the *BEA_HOME*`\user_projects\domains\portalDomain` directory, and double-click the `create_db.cmd` file.
   b. For UNIX, navigate to the *BEA_HOME*`\user_projects\domains\portalDomain` directory and run `create_db.sh`.
   c. Verify the results in the `create_db.log` file.

Note: If you are using the sample domain, run the create_db.cmd/sh file from the following directory: *WL_HOME*`\samples\domains\portal`.

5. Follow the steps in "Manually Configuring Your Domain's JDBC Driver Settings" on page 5-6.

Manually Configuring Your Domain's JDBC Driver Settings

You can either manually configure your domain's JDBC driver settings using the WebLogic Server Console, or use the Configuration Wizard; for more information, see "Overview of Database Configuration for WebLogic Portal" on page 1-1.

To manually configure your JDBC driver settings using WebLogic Server Console:

1. Start the WebLogic Server for your domain.
2. Login to the WebLogic Server Console.
3. Configure your new connection pools.
   a. Go to Services →JDBC →Connection Pools.
   b. Click Configure a New Connection Pool.
   c. Select the appropriate database type and non-XA database driver from the drop-down lists and click Continue. For more information, see the Supported Configuration documentation for JDBC drivers supported by WebLogic Portal located at http://edocs.bea.com/platform/docs81/support/supp_plat.html.

      For an XA configuration, see "Creating XA Domains Using Configuration Templates" in the "Creating WebLogic Configurations Using the Configuration Wizard documentation located at http://edocs.bea.com/platform/docs81/confgwiz/index.html.
   d. Choose a name for the new connection pool (for example, cgPoolN) and fill in the blanks for your vendor database. Click Continue.
   e. Test your connection to verify that you can successfully connect to your database.
   f. Create and deploy your new connection pool.

      Note: You must maintain a one-to-one mapping of JDBCTxDataSource to JDBC connection pool in the domain's `config.xml` file. Create one new JDBC connection pool for each JDBCTxDataSource and another JDBC connection pool for the domain's JDBCDataSources.
4. Update your data sources.
   a. From Services →JDBC →Data Sources, click each data source and switch each to the newly created connection pool. Make sure that you apply each change.
   b. Verify that each data source is changed by clicking on Data Sources and then verifying that Pool Name has been set to the new connection pool for each.

5. From Services →JMS →Stores →cgJMSStore, switch cgJMSStore to use the new connection pool.
6. Stop your domain's WebLogic Server, then restart it.
7. In the WebLogic Server Console, delete the original connection pools.
   a. Go to Services →JDBC →Connection Pools.
   b. Right-click each connection pool and select Delete.
8. This step is recommended for improved performance. Move indexes to the WEBLOGIC_INDEX tablespace by executing rebuild_indexes.sql from SQLPLUS. Do this while WebLogic Server is not running.

Creating a Separate Database for Behavior Tracking Events

For improved performance, you might want to store behavior tracking events in a different location from other WebLogic Portal database objects. For more information about behavior tracking, see http://e-docs.bea.com/wlp/docs81/adminportal/help/SA_BehavTrackServ.html.

Note: By default, behavior tracking database objects are created in the same database as other WebLogic Portal database objects. You need to perform these steps only if you are configuring a separate database for behavior tracking events.

1. Edit the `bt_create_tablespaces.sql` file and the `bt_create_users.sql` file for your environment, as indicated in the instructions contained in the files. .
2. From SQL Plus, run the `bt_create_tablespaces.sql` script.
3. From SQL Plus, run the `bt_create_users.sql` script.
4. Navigate to the appropriate database directory based on your environment: *`WL_HOME`*`\portal\db\oracle\817`
5. Connect as the user `WEBLOGIC_EVENT` and run the following scripts:
   - `bt_create_tables.sql`
   - `bt_create_fkeys.sql`
   - `bt_create_indexes.sql`
   - `bt_create_views.sql`
   - `bt_create_triggers.sql`
6. Run the following script from the path *`WL_HOME`*`\portal\db\data\required`:

```
– bt_insert_system_data.sql
```

7. Configure a connection pool to access your behavior tracking database and associate the p13n_tracking JDBC data source with that connection pool. Follow the steps in "Manually Configuring Your Domain's JDBC Driver Settings" on page 5-6.

WebLogic Platform Support for Oracle9i RAC

WebLogic Platform 8.1 SP4 now provides a patch that enables WebLogic Platform 8.1 SP4 support for Oracle9i Real Application Clusters (RAC). For more information about this patch, see the WebLogic Platform Release Notes at the following URL:

http://edocs.bea.com/platform/docs81/relnotes/relnotes.html

Oracle 10g R2 Database Requires Manual Privilege Setup

As described in Oracle Note 317258.1, Oracle changed their use of predefined database roles in the 10g R2 version to provide more security. The CONNECT role now provides only CREATE SESSION privileges; the other associated privileges have been removed.

You must now explicitly grant the CREATE VIEW privilege for Oracle 10g R2 databases.

CHAPTER 6

Using a Sybase Database

This section describes the steps necessary to use a Sybase database with WebLogic Portal 8.1, and includes information on the following subjects:

- Configuring a Sybase Database
- Manually Creating Database Objects
- Manually Configuring Your Domain's JDBC Driver Settings
- Creating a Separate Database for Behavior Tracking Events

Review this entire chapter and any release notes before proceeding. The steps in this chapter should be performed by a database administrator.

Note: For additional database setup information, see "Managing WebLogic Platform Database Resources" at http://e-docs.bea.com/platform/docs81/db_mgmt/db_resource_mgmt.html.

Configuring a Sybase Database

Before proceeding, be sure that you have read "Overview of Database Configuration for WebLogic Portal" on page 1-1.

The following sections contain special considerations for defining your Sybase instance, as well as setup instructions.

Using a Supported Version

Be sure that you are using a supported version; see http://edocs.bea.com/platform/docs81/support/supp_plat.html#1085671.

Defining an 8K Page Size

You must define a page size of at least 8K to support WebLogic Portal's use of wide tables, wide columns, and larger indexes. An 8K page size is the default for most databases. However, for Sybase the default page size is 2K, and Sybase does not allow rows to span pages.

If a Sybase instance is defined with a page size smaller than 8K, the WebLogic Portal tables will be created but warning messages might occur at creation time, indicating that the row size could exceed the row size limit. These warnings may result in run time exceptions, depending on the data being inserted or updated.

Indexes will fail to create if they are larger than the maximum page size for the Sybase instance. This could result in data issues as well as performance problems.

If your Sybase instance uses 2k or 4k pages, create a new Sybase instance with an 8K page size. Sybase provides a migration utility to migrate data between servers of different page sizes. You can find a technical white paper on the Sybase migration process at http://www.sybase.com/detail/printthis/1,6907,1021203,00.html.

Running Upgrade Script for 7.0 to 8.1

For WebLogic Portal 7.0 users who are upgrading to Version 8.1, ensure that the following WebLogic Portal 7.0 script has been run:

`bea\weblogic700\portal\db\sybase\125\migrate\migrate_to_125.sql`

Install and Configure the Sybase Client

1. Install the Sybase client software on the WebLogic Platform host and do the following:
   a. Configure the client so that it connects to the target Sybase instance.
   b. Verify that you can connect to the target instance using `isql`. For example,

      ```
      isql -Usa -Ppassword -SMySybase
      ```

2. Prepare the Sybase database. The database creation scripts install domain-specific tables. You should work with your database administrator to adjust the sample scripts, and to create the database schema owner users and devices needed for your environment.

Notes: Multiple databases are required if you have multiple domains, or to run multiple environments using the same Sybase instance (for example, if you want to run your test environment and production system from a single Sybase installation).

Be sure to back up your database(s) before installing any new database objects. See your database documentation for details.

a. Review and modify the provided sample scripts to suit your environment. These scripts are provided in the *WL_HOME*/portal/db/sybase/125/admin directory.

Script Name	Description
create_devices.sql	Creates database devices. **Usage Notes:** Database devices must be created by a user with system administrator privileges (normally the `sa` user). `D:\DATAFILE`, `E:\LOGFILE`, and `F:\INDEXFILE` specifications in this script must be changed to reflect valid disk locations for your environment. Optimally, data, log, and index devices would be placed on separate physical disks that reside on separate controllers, unless you are using RAID devices. Edits are required to change file sizes and device names. The following default names are used: • data device: `WEBLOGIC_DATA` • log device: `WEBLOGIC_LOG` • index device: `WEBLOGIC_INDEX`
create_database.sql	Creates the database and login. An alias is added to the database owner (dbo) user of the database. The devices created by `create_devices.sql` are used and a `WEBLOGIC_INDEX` is added. **Usage Notes:** Edit the script to reflect name or size changes from create_devices.sql. Edits are required to change the default database name and/or dbo user. The following defaults are used: • data device: `WEBLOGIC_DATA` • log device: `WEBLOGIC_LOG` • database name: `WEBLOGIC` • database owner user: `WEBLOGIC` • password: `WEBLOGIC` If the database you are creating is a development database, your database administrator might want to uncomment and set the `truncate log on checkpoint` database option. If your application will use WebLogic Workshop page flows or RowSet controls, uncomment and set the `DDL in transaction` option to true to allow database table `create` commands to work properly.

Script Name	Description
statistics_build.sql	Builds `statistics.sql` to update table and index statistics for the database optimizer. Update statistics whenever any significant changes in database data occur. Your database administrator should schedule update statistics to run periodically in your environment.
install_report_build.sql install_report_static.sql	Builds an informational installation report about the database objects created by the `WEBLOGIC` user.
bt_create_devices.sql	Creates behavior tracking database devices. **Usage Notes:** Database devices must be created by a user with system administrator privileges (normally the `sa` user). `D:\DATAFILE` and `E:\LOGFILE` specifications in this script must be changed to reflect valid disk locations for your environment. Optimally, data and log devices would be placed on separate physical disks that reside on separate controllers. Edits are required to change file sizes and device names. The following default names are used: • data device: WEBLOGIC_EVENT_DATA • log device: WEBLOGIC_EVENT_LOG
bt_create_database.sql	Create the WEBLOGIC_EVENT database and WEBLOGIC_EVENT database owner user login. An alias is created to make WEBLOGIC_EVENT the database owner (dbo) user in the database. **Usage Notes:** Edit the script to change database names, the dbo user, and password. Edits are required to reflect valid disk locations for DATA and the LOG devices, or to adjust file sizes. Put DATA and LOG files on separate physical disks and away from any system database files. The following defaults are used: • data device: WEBLOGIC_EVENT_DATA • log device: WEBLOGIC_EVENT_LOG • database name: WEBLOGIC • database owner user: WEBLOGIC • password: WEBLOGIC

b. Run `create_devices.sql` as a user with system administrator privileges. For example:
```
isql -Usa -SMYSYBASE -e -icreate_devices.sql -ocreate_devices.log
```

c. Run `create_database.sql` using isql as a user with System Administrator privileges (that is, the `sa` user):

```
isql -Usa -SMYSYBASE -e -icreate_database.sql -ocreate_database.log
```

Output is written to the file specified after the `-o` parameter. The log file is stored in the same directory in which the script resides. Verify that each log file contains no errors for database object creation.

d. Statistics and install report scripts are run automatically by the `create_db.cmd/.sh` scripts. Ensure that your database administrator schedules statistics updates to run periodically for your WebLogic Portal database.

3. Follow the steps in "Manually Creating Database Objects" on page 6-6.

Manually Creating Database Objects

You can either manually create database objects or use the Configuration Wizard. For more information, see "Overview of Database Configuration for WebLogic Portal" on page 1-1.

Note: If you choose to use the WebLogic Configuration Wizard to configure and connect to the database that you will use to support WebLogic Portal, see http://edocs.bea.com/platform/docs81/confgwiz/index.html.

To manually create WebLogic Portal database objects, use the following steps:

1. Verify that you can connect to the target database. Use the following command syntax to verify that you can connect to the target database server using the default schema owner user created by running `create_database.sql`.

```
isql -UWEBLOGIC -SMYSYBASE
```

2. Open your domain's `db_settings.properties` file for edit and comment out the database setting for PointBase.
3. Uncomment the database settings for your new target database and update the following settings for your database:
   - `server=`
   - `dblogin=`
   - `password=`
4. Initialize the database with the new settings.

a. Navigate to the *BEA_HOME*\user_projects\domains\portalDomain directory, and double-click the create_db.cmd file.

b. Verify the results in the create_db.log file.

Note: If you are using the sample domain, run the create_db.cmd/sh file from the following directory: *WL_HOME*\samples\domains\portal.

5. Follow the steps in "Manually Configuring Your Domain's JDBC Driver Settings" on page 6-7.

Manually Configuring Your Domain's JDBC Driver Settings

You can either manually configure your domains JDBC driver settings using the WebLogic Server Console, or use the Configuration Wizard, see "Manually Creating Database Objects and JDBC Settings" on page 1-2 for more information.

To manually configure your JDBC driver settings using WebLogic Server Console:

1. Start the WebLogic Server for your domain.
2. Log on to the WebLogic Server Console.
3. Configure your new connection pools.

   a. Go to Services →JDBC →Connection Pools.

   b. Click Configure a new Connection Pool.

   c. Select the appropriate database type and non-XA database driver from the drop-down list boxes and click Continue. For more information, see the Supported Configuration documentation for JDBC drivers supported by WebLogic Portal located at http://edocs.bea.com/platform/docs81/support/supp_plat.html#1085671.

      For an XA configuration, see "Creating XA Domains Using Configuration Templates" in the "Creating WebLogic Configurations Using the Configuration Wizard documentation, http://edocs.bea.com/platform/docs81/confgwiz/index.html.

   d. Choose a name for the new connection pool (for example, cgPoolN) and fill in the blanks for your vendor database. Click Continue.

   e. Test your connection to verify that you can successfully connect to your database.

   f. Create and deploy your new connection pool.

Note: You must maintain a one-to-one mapping of JDBCTxDataSource to JDBC connection pool in the domain's `config.xml` file. Create one new JDBC connection pool for each JDBCTxDataSource and another JDBC connection pool for the domain's JDBCDataSources.

4. Update your data sources.
   a. From Services →JDBC →Data Sources, click each data source and switch each to the newly created connection pool. Be sure to apply each change.
   b. Verify that each data source is changed by clicking on Data Sources and then verifying that Pool Name has been set to the new connection pool for each.
5. From Services →JMS →Stores →cgJMSStore, switch cgJMSStore to use the new connection pool.
6. Stop your domain's WebLogic Server, then restart it.
7. In the WebLogic Server Console, delete the original connection pools.
   a. Go to Services →JDBC →Connection Pools.
   b. Right-click each connection pool and select Delete.

Creating a Separate Database for Behavior Tracking Events

For improved performance, you might want to store behavior tracking events in a different location from other WebLogic Portal database objects. For more information about behavior tracking, see http://e-docs.bea.com/wlp/docs81/adminportal/help/SA_BehavTrackServ.html.

Note: By default, behavior tracking database objects are created in the same database as other WebLogic Portal database objects. You need to perform these steps only if you are configuring a separate database for behavior tracking events.

1. Edit the `bt_create_devices.sql` file and the `bt_create_database.sql` file for your environment, as indicated in the instructions contained in the files. .
2. Run `bt_create_devices.sql` using isql as a user with system administrator privileges. For example:

```
isql -Usa -SMYSYBASE -e -ibt_create_devices.sql
-obt_create_devices.log
```

3. Run `bt_create_database.sql` using isql as a user with system administrator privileges. For example:

```
isql -Usa -SMYSYBASE -e -ibt_create_database.sql
-obt_create_database.log
```

4. Navigate to the appropriate database directory based on your environment: *WL_HOME*\portal\db\sybase\125
5. Connect as the user WEBLOGIC_EVENT and run the following scripts:
   - bt_create_tables.sql
   - bt_create_fkeys.sql
   - bt_create_indexes.sql
   - bt_create_views.sql
   - bt_create_triggers.sql
6. Run the following script from the path *WL_HOME*\portal\db\data\required:
   - bt_insert_system_data.sql
7. Configure a connection pool to access your behavior tracking database and associate the p13n_tracking JDBC data source with that connection pool. Follow the steps in "Manually Configuring Your Domain's JDBC Driver Settings" on page 6-7.

CHAPTER 7

Using a DB2 Database

This section describes the steps necessary to use a DB2 database with WebLogic Portal 8.1, and includes information on the following subjects:

- Configuring a DB2 Database
- Manually Creating Database Objects
- Manually Configuring Your Domain's JDBC Driver Settings
- Creating a Separate Database for Behavior Tracking Events

Typically, the steps in this chapter should be performed by a database administrator.

Review this entire chapter and any release notes before proceeding.

Note: For additional database setup information, see "Managing WebLogic Platform Database Resources" at http://e-docs.bea.com/platform/docs81/db_mgmt/db_resource_mgmt.html.

Configuring a DB2 Database

Before proceeding, be sure that you have read "Overview of Database Configuration for WebLogic Portal" on page 1-1.

1. Install the DB2 client software and configure it to connect to the target DB2 database. See your DB2 documentation for more information.
2. Verify that you can connect to the target database through the Command Line Processor (CLP).

3. Prepare the DB2 database. The database creation scripts install domain-specific tables for each. It is recommended that you work with a database administrator to adjust the sample scripts, and to create the database objects (users, passwords, tablespaces, and so on) needed for your environment.

Notes: Multiple database schemas are required if you have multiple domains, or to run multiple environments using the same DB2 instance (for example, if you want to run development and system test from a single DB2 installation).

Be sure to back up your database before installing any new database objects. See your database documentation for details.

DB2 Configuration Parameter Minimum Settings

To ensure that the Portal application can successfully run on DB2, you must set some minimum configuration parameters. Without the minimum settings, heavy Portal activity might exceed database capacity.

Use the following minimum settings as a guideline as you configure your DB2 database:

- Dynamic Sections: 20,000
- applheapsz: 24,000
- pckcachesz: 2,500

a. Review and modify the provided sample scripts to suit your environment. The scripts are located in *WL_HOME*/portal/db/db2/8/admin.

The following table lists the script names and the usage notes for each script.

Script Name	Description
create_user.sql	Grants createtab, bindadd and connect DB2 privileges to the WEBLOGIC schema owner user. **Usage Notes:** Because IBM DB2 databases authenticate users using the operating system (OS), you need to create an OS user that owns database schema objects. Edit the script to change the schema owner user name. The default schema owner user name and password are the following: • schema owner user: WEBLOGIC • schema owner user password: WEBLOGIC
create_bufferpool.sql	Creates an 8K bufferpool, if one does not already exist. **Usage Notes:** You must stop and restart DB2 to utilize new bufferpools. Edit the script to change the 8K bufferpool name. The default bufferpool is BP8K.
create_tablespaces.sql	Creates 4K and 8K regular tablespaces. The default tablespace names are the following: WEBLOGIC_DATA_4K: Tables for WebLogic Portal and/or WebLogic Platform with a rowsize smaller than 4K. WEBLOGIC_DATA_8K: Tables for WebLogic Portal and/or WebLogic Platform with a rowsize larger than 4K and smaller than 8K. **Usage Notes**: Edit the script to specify valid physical disk locations for your environment (d:\db2\data\data4k), for a database user other than WEBLOGIC and to change buffer pool names.
create_temp_tablespaces.sql	Creates an 8K temporary tablespace. The default tablespace name is TEMPSPACE_8K. **Usage Notes:** Edit the script to specify valid physical disk locations for your environment (d:\db2\data\data4k), for a database user other than WEBLOGIC and to change buffer pool names.

Script Name	Description
statistics_build.sql	Builds a file of `runstats` commands for each table that will compute database statistics needed for the database optimizer. Run `runstats` whenever any significant changes in database data occur. Your database administrator typically schedules `runstats` to run periodically in your environment.
install_report.sql	Builds an informational installation report about the database objects created in the WEBLOGIC schema.
bt_create_tablespace.sql	Creates the WEBLOGIC_EVENT_DATA tablespace. **Usage Notes:** Edit the script to specify valid physical disk locations for your environment (*`event_container`*), and to use a buffer pool other than the IBM default buffer pool.
bt_create_users.sql	Creates the WEBLOGIC_EVENT schema owner user, establishes the user's password, default and temporary tablespaces, and grants privileges to that user. **Usage Notes:** Edit the script to change the schema owner user name, password and tablespace names. The default schema owner user name and password are the following: • schema owner user: WEBLOGIC • schema owner user password: WEBLOGIC

b. Start the CLP DB2 tool from the directory that contains the scripts.

c. From CLP, connect to the database that you want to work with. For example, type:

```
Db2 connect to database user username password password
```

d. From CLP, run `create_bufferpool.sql`, if needed. You might not need to create a new 8K bufferpool if you already have one to use. For example:

```
Db2 -tf create_bufferpool.sql -v
```

e. Restart your database instance.

f. From CLP, run `create_temp_tablespaces.sql`. For example:

```
Db2 -tf create_temp_tablespaces.sql -v
```

g. From CLP, run `create_tablespaces.sql`. For example:

```
Db2 -tf create_tablespaces.sql -v
```

h. From CLP, run `create_user.sql`. For example:

```
Db2 -tf create_user.sql -v
```

4. Follow the steps in "Manually Creating Database Objects" on page 7-5.

Manually Creating Database Objects

You can either manually create database objects or use the Configuration Wizard. For more information, see "Overview of Database Configuration for WebLogic Portal" on page 1-1.

Note: If you choose to use the WebLogic Configuration Wizard to configure and connect to the database that you will use to support WebLogic Portal, see http://edocs.bea.com/platform/docs81/confgwiz/index.html.

To manually create BEA Portal database objects, use the following steps:

1. From DB2-CLP, use the following command to verify that you can connect to the target database server with a valid user ID and password:

   ```
   db2 connect to database user username password password
   ```

2. Open your domain's `db_settings.properties` file for edit and comment out the database settings for PointBase.
3. In the `db_settings.properties` file for your domain, uncomment the database settings for your new target database and update the following settings for your database:
   - `server=`
   - `dblogin=`
   - `password=`
4. Create the database.
   a. For Windows, navigate to the *BEA_HOME*`\user_projects\domains\portalDomain` directory, and double-click the create_db.cmd file.
   b. For UNIX, navigate to the *BEA_HOME*`\user_projects\domains\portalDomain` directory, run create_db.sh.
   c. Verify the results in the `create_db.log` file.

Note: If you are using the sample domain, run the create_db.cmd/sh file from the following directory: `WL_HOME\samples\domains\portal`.

5. Follow the steps in "Manually Configuring Your Domain's JDBC Driver Settings" on page 7-6.

Manually Configuring Your Domain's JDBC Driver Settings

You can either manually configure your domains JDBC driver settings using the WebLogic Server Console, or use the Configuration Wizard. For more information., see "Overview of Database Configuration for WebLogic Portal" on page 1-1.

To manually configure your JDBC driver settings using WebLogic Server Console:

1. Start the WebLogic Server for your domain.
2. Login to the WebLogic Server Console.
3. Configure your new connection pools.
   a. Go to Services →JDBC →Connection Pools.
   b. Click Configure a new Connection Pool.
   c. Select the appropriate database type and non-XA database driver from the drop-down list boxes and click Continue. For more information, see the Supported Configuration documentation for JDBC drivers supported by WebLogic Portal located at http://edocs.bea.com/platform/docs81/support/supp_plat.html#1085671.

      For an XA configuration, see "Creating XA Domains Using Configuration Templates" in the "Creating WebLogic Configurations Using the Configuration Wizard documentation, http://edocs.bea.com/platform/docs81/confgwiz/index.html.
   d. Choose a name for the new connection pool (for example, cgPoolN) and fill in the blanks for your vendor database. Click Continue.
   e. Test your connection to verify that you can successfully connect to your database.
   f. Create and deploy your new connection pool.

      Note: You must maintain a one-to-one mapping of JDBCTxDataSource to JDBC connection pool in the domain's `config.xml` file. Create one new JDBC connection pool for each JDBCTxDataSource and another JDBC connection pool for the domain's JDBCDataSources.
4. Update your data sources.
   a. From Services →JDBC →Data Sources, click each data source and switch each to the newly created connection pool. Be sure to apply each change.
   b. Verify that each data source is changed by clicking on Data Sources and then verifying that Pool Name has been set to the new connection pool for each.

5. From Services →JMS →Stores →cgJMSStore, switch cgJMSStore to use the new connection pool.
6. Stop your domain's WebLogic Server, then restart it.
7. In the WebLogic Server Console, delete the original connection pools.
   a. Go to Services →JDBC →Connection Pools.
   b. Right-click each connection pool and select Delete.

Creating a Separate Database for Behavior Tracking Events

For improved performance, you might want to store behavior tracking events in a different location from other WebLogic Portal database objects. For more information about behavior tracking, see http://e-docs.bea.com/wlp/docs81/adminportal/help/SA_BehavTrackServ.html.

Note: By default, behavior tracking database objects are created in the same database as other WebLogic Portal database objects. You need to perform these steps only if you are configuring a separate database for behavior tracking events.

1. Edit the `bt_create_tablespaces.sql` file and the `bt_create_users.sql` file for your environment, as indicated in the instructions contained in the files. .
2. From CLP, run the `bt_create_tablespaces.sql` script. For example, type:

   ```
   Db2 -tf bt_create_tablespaces.sql -v
   ```

3. From CLP, run the `bt_create_users.sql` script. For example, type:

   ```
   Db2 -tf bt_create_users.sql -v
   ```

4. Navigate to the appropriate database directory based on your environment: *`WL_HOME`*`\portal\db\db2\8`
5. Connect as the user `WEBLOGIC_EVENT` and run the following scripts:
   - `bt_create_tables.sql`
   - `bt_create_fkeys.sql`
   - `bt_create_indexes.sql`
   - `bt_create_views.sql`
   - `bt_create_triggers.sql`
6. Run the following script from the path *`WL_HOME`*`\portal\db\data\required`:
   - `bt_insert_system_data.sql`

7. Configure a connection pool to access your behavior tracking database and associate the p13n_tracking JDBC data source with that connection pool. Follow the steps in "Manually Configuring Your Domain's JDBC Driver Settings" on page 7-6.

CHAPTER 8

Data Dictionary

This section describes the database objects for each component of WebLogic Portal. The information in this section is collectively known as the data dictionary.

Information Provided

For each component of WebLogic Portal, the following information is provided:

- An entity-relationship diagram
- A detailed description of each database table, including:

  Table Name
  : The predefined name for the Table.

  Table Description
  : A detailed description of the contents and purpose for the table in WebLogic Portal database schema.

  Column Name
  : The predefined name for the column.

  Data Type
  : The predefined characteristics for the column.

  Note: Data types vary slightly by DBMS. For instance, columns defined as BLOB data types in Oracle, DB2, and PointBase would be defined as TEXT columns in Microsoft SQL Server and Sybase.

Null Value
: Indicates whether or not null values can be stored for the column.

Column Description
: A detailed description of the contents and purpose for the column including Primary Key (PK-) and Foreign Key (FK-) designations.

Note: The term "hint" in the descriptions refers to available capabilities that are not supported in the default skeletons provided with the WebLogic Workshop Portal Extensions

Portal Database Components Covered

This section includes information on the following subjects:

- Behavior Tracking Database Objects
- Commerce Services Database Objects
- Order and Discount Database Objects
- Personalization Database Objects
- Data Synchronization Database Objects
- WebLogic Portal Services Database Objects
- Portal Framework Database Objects
  - WSRP (Web Services for Remote Portlets) Objects
- Content Management Database Objects
- Content Management Virtual Database Objects
- Localization Database Objects
- Tracked Anonymous User Database Objects
- Entitlement Reference Database Objects

Note: Appendix A, "WebLogic Portal DDL Modules" identifies the filenames and location of DDL (database definition language) files for each set of Portal database objects.

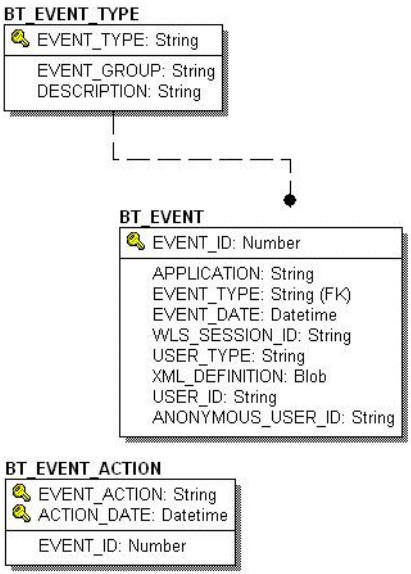
Behavior Tracking Database Objects

To record how online visitors are interacting with your Web site, you can record event information to a database. These kinds of events are called behavior tracking events. Analytics

Marketing systems can then analyze these events offline to evaluate visitor behavior and transactional data. You can use the knowledge gained from analysis to create and optimize personalization rules, set up product offers, and develop interactive marketing campaigns. This section describes the requirements and database objects needed to log event data for analytical use.

Three tables are provided for the behavior tracking data. The BT_EVENT table stores all event data. The BT_EVENT_ACTION table logs actions used by third-party vendors against the recorded event data, and the BT_EVENT_TYPE table references event types and categories in the EVENT table. Figure 8-1 shows an entity-relation diagram for the WebLogic Portal Behavior Tracking database objects.

Figure 8-1 Entity-Relation Diagram for the Behavior Tracking Database



The BT_EVENT_TYPE Database Table

This table references event types and categories in the BT_EVENT table. This table is static.

Table 8-1 BT_EVENT_TYPE Table Metadata

Column Name	Data Type	Null Value	Description
EVENT_TYPE	VARCHAR(30)	Not Null	PK - A unique, system-generated number used as the record ID.
EVENT_GROUP	VARCHAR(10)	Not Null	The event category group associated with the event type.
DESCRIPTION	VARCHAR(50)	Null	A description of the EVENT_TYPE.

To record custom events, you must create an entry in this table. If a custom event does not have a record in this table, you cannot persist it to the BT_EVENT table.

The BT_EVENT Database Table

This table stores all behavior tracking event data.

Table 8-2 The BT EVENT Table Metadata

Column Name	Data Type	Null Value	Description
EVENT_ID	NUMBER	Not Null	PK - A unique, system-generated number used as the record ID.
APPLICATION	VARCHAR (30)	Not Null	The application that created the event.
EVENT_TYPE	VARCHAR(30)	Not Null	FK - Set to BT_EVENT_TYPE. A string identifier showing which event was fired.
EVENT_DATE	DATE	Not Null	The date and time of the event.
WLS_SESSION_ID	VARCHAR(254)	Not Null	A unique, WebLogic Server-generated number assigned to the session.
XML_DEFINITION	CLOB	Null	An XML document that contains the specific event information for each event type. It is stored as a CLOB (Character Large Object). See Table 8-3.

Table 8-2 The BT EVENT Table Metadata (Continued)

Column Name	Data Type	Null Value	Description
USER_ID	VARCHAR(50)	Null	The user ID associated with the session and event. If the user has not logged in this column is null.
ANONYMOUS_USER_ID	VARCHAR(128)	Null	The user ID of the anonymous user associated with the session and event, if applicable.

As shown in Table 8-2, the BT_EVENT table has six columns; each column corresponds to a specific event element. Five of the BT_EVENT table's columns contain data common to every event type. The XML_DEFINITION column contains all information from these five columns plus event data that is unique to each event type. An XML document is created specifically for each event type. The data elements corresponding to each event type are captured in the XML_DEFINITION column of the EVENT table. These elements are listed in Table 8-3.

Table 8-3 XML_DEFINITION Data Elements

Event	Data Element
AddToCartEvent	application event-date event-type session-id user-id sku quantity unit-list-price currency application-name

Table 8-3 XML_DEFINITION Data Elements (Continued)

Event	Data Element
BuyEvent	application event-date event-type session-id user-id sku quantity unit-price currency application-name order-line-id
CampaignUserActivityEvent	application event-date event-type session-id user-id campaign-id scenario-id
ClickCampaignEvent	application event-date event-type session-id user-id document-type document-id campaign-id scenario-id application-name placeholder-id
ClickContentEvent	application event-date event-type session-id user-id document-type document-id

Table 8-3 XML_DEFINITION Data Elements (Continued)

Event	Data Element
ClickProductEvent	application event-date event-type session-id user-id document-type document-id sku category-id application-name
DisplayCampaignEvent	application event-date event-type session-id user-id document-type document-id campaign-id scenario-id application-name placeholder-id
DisplayContentEvent	application event-date event-type session-id user-id document-type document-id
DisplayProductEvent	application event-date event-type session-id user-id document-type document-id sku category-id application-name

Table 8-3 XML_DEFINITION Data Elements (Continued)

Event	Data Element
PurchaseCartEvent	application event-date event-type session-id user-id total-price order-id currency application-name
RemoveFromCartEvent	application event-date event-type session-id user-id sku quantity unit-price currency application-name
RuleEvent	application event-date event-type session-id user-id ruleset-name rule-name
SessionBeginEvent	application event-date event-type session-id user-id
SessionEndEvent	application event-date event-type session-id user-id

Table 8-3 XML_DEFINITION Data Elements (Continued)

Event	Data Element
SessionLoginEvent	application event-date event-type session-id user-id
UserRegistrationEvent	application event-date event-type session-id user-id

The BT_EVENT_ACTION Database Table

This table logs actions used by third-party vendors against the recorded event data.

Table 8-4 BT_EVENT_ACTION Table Metadata

Column Name	Data Type	Null Value	Description
EVENT_ACTION	VARCHAR(30)	Not Null	The event action taken such as BEGIN EXPORT or END EXPORT. This field is one of the table's primary keys.
ACTION_DATE	DATE	Not Null	The date and time of the event. This field is one of the table's primary keys.
EVENT_ID	NUMBER	Null	The ID of the event that corresponds with the event action taken.

Commerce Services Database Objects

The metadata for items in the Commerce Services Product Catalog are based on the Dublin Core Metadata Open Standard. This standard offers a number of advantages for a Web-based catalog. For more information about the Dublin Core Metadata Open Standard, please see http://dublincore.org.

Figure 8-2 and Figure 8-3 show the Entity-Relation for the WebLogic Portal Commerce Services core Product Catalog database objects.

Figure 8-2 Entity-Relation Diagram for the Core Product Catalog Tables

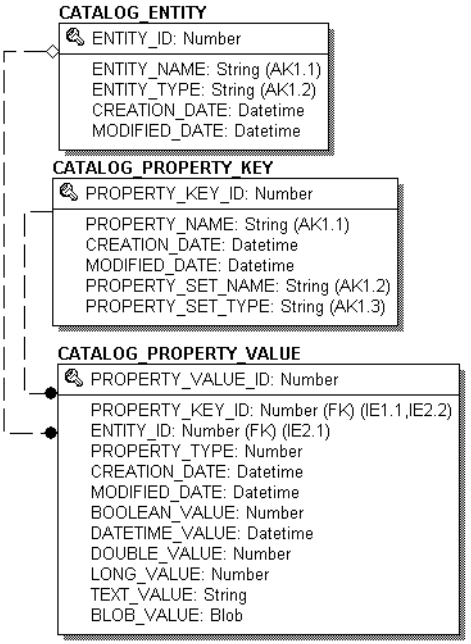
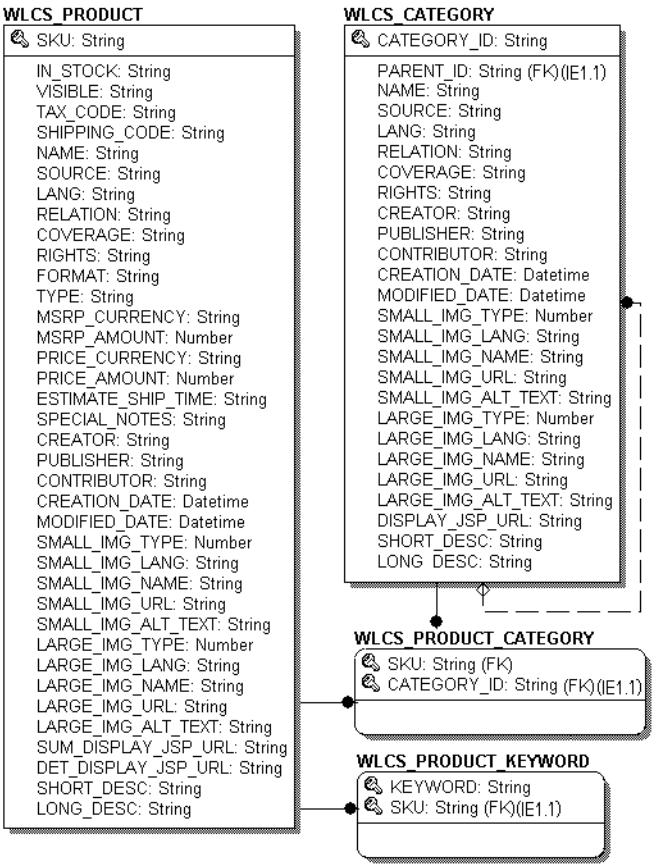


Figure 8-3 Entity-Relation Diagram for the Core Product Catalog Tables - continued



Product Catalog Database Tables

The following tables compose the product catalog database.

- The CATALOG_ENTITY Database Table
- The CATALOG_PROPERTY_KEY Database Table
- The CATALOG_PROPERTY_VALUE Database Table
- The WLCS_CATEGORY Database Table
- The WLCS_PRODUCT Database Table

- The WLCS_PRODUCT_CATEGORY Database Table
- The WLCS_PRODUCT_KEYWORD Database Table

The CATALOG_ENTITY Database Table

Some objects in WebLogic Portal implement a Java interface called ConfigurableEntity. Any ConfigurableEntity within the system has an entry in this table.

Table 8-5 CATALOG_ENTITY Table Metadata

Column Name	Data Type	Null Value	Description
ENTITY_ID	NUMBER(15)	Not Null	PK—A unique, system-generated number used as a record identifier.
ENTITY_NAME	VARCHAR(200)	Not Null	The name of the entity.
ENTITY_TYPE	VARCHAR(100)	Not Null	The type of entity; for example, User, Group, and so on.
CREATION_DATE	DATE	Not Null	The time and date the record was created.
MODIFIED_DATE	DATE	Not Null	The time and date the record was last modified.

The CATALOG_PROPERTY_KEY Database Table

This table contains scoped property names that are associated with configurable entities. Any property assigned to a ConfigurableEntity has a unique PROPERTY_ID. This identifier and associated information is stored here.

Table 8-6 CATALOG_PROPERTY_KEY Table Metadata

Column Name	Data Type	Null Value	Description
PROPERTY_KEY_ID	NUMBER(15)	Not Null	PK—A unique, system-generated number used as a record identifier.
PROPERTY_NAME	VARCHAR(100)	Not Null	The name of the property (formerly PROPERTY_NAME from the WLCS_PROP_ID table).
CREATION_DATE	DATE	Not Null	The time and date the record was created.

Table 8-6 CATALOG_PROPERTY_KEY Table Metadata (Continued)

Column Name	Data Type	Null Value	Description
MODIFIED_DATE	DATE	Not Null	The time and date the record was last modified.
PROPERTY_SET_NAME	VARCHAR(100)	Null	The name of the property set (formerly the SCOPE_NAME from WLCS_PROP_ID).
PROPERTY_SET_TYPE	VARCHAR(100)	Null	The type of property set (for example, USER)

The CATALOG_PROPERTY_VALUE Database Table

This table contains boolean, timestamp, float, integer, text, and user-defined (object) property values that are associated with configurable entities.

Table 8-7 CATALOG_PROPERTY_VALUE Table Metadata

Column Name	Data Type	Null Value	Description
PROPERTY_VALUE_ID	NUMBER(15)	Not Null	PK—A unique, system-generated number used as a record identifier.
PROPERTY_KEY_ID	NUMBER(15)	Not Null	FK—A system-generated value and foreign key to the PROPERTY_KEY column.
ENTITY_ID	NUMBER(15)	Not Null	FK—A system-generated value and foreign key to the ENTITY column.
PROPERTY_TYPE	NUMBER(1)	Not Null	Valid entries are: 0=Boolean, 1=Integer, 2=Float, 3=Text, 4=Date and Time, 5=User-Defined (BLOB).
CREATION_DATE	DATE	Not Null	The time and date the record was created.
MODIFIED_DATE	DATE	Not Null	The time and date the record was last modified.
BOOLEAN_VALUE	NUMBER(1)	Null	The value for each boolean property identifier.

Table 8-7 CATALOG_PROPERTY_VALUE Table Metadata (Continued)

Column Name	Data Type	Null Value	Description
DATETIME_VALUE	DATE	Null	The value for each date and time property identifier.
DOUBLE_VALUE	NUMBER	Null	The value associated with each float property identifier.
LONG_VALUE	NUMBER(20)	Null	The value associated with the integer property.
TEXT_VALUE	VARCHAR(254)	Null	The value associated with the text property.
BLOB_VALUE	BLOB	Null	The value associated with the user-defined property.

The WLCS_CATEGORY Database Table

This table contains categories in the Commerce database. The descriptions shown in the table reflect the "recommended best practice" for the use of that field by the Dublin Core standard.

Table 8-8 WLCS_CATEGORY Table Metadata

Column Name	Data Type	Null Value	Descriptions
CATEGORY_ID	VARCHAR(20)	Not Null	PK - A unique identifier for a category; the primary key for this table. This field cannot be NULL. All other fields in the WLCS_CATEGORY table can be NULL.
PARENT_ID	VARCHAR(20)	Null	The value of the CATEGORY_ID of the parent category in the hierarchy of categories that comprise your product catalog. If this is a top-level user-defined category, the PARENT_ID is com.beasys.ROOT.
NAME	VARCHAR(50)	Null	The name of the category in the product catalog.
SOURCE	VARCHAR(30)	Null	A reference to a category from which the present category is derived.

Table 8-8 WLCS_CATEGORY Table Metadata (Continued)

Column Name	Data Type	Null Value	Descriptions
LANG	VARCHAR(30)	Null	A language of the intellectual content of the category. The recommended best practice for the values of the language element is defined by RFC 1766, which includes a two-letter Language Code (taken from the ISO 639 standard), such as: en for English; fr for French, or de for German. The language code can, optionally, be followed by a two-letter Country Code (taken from the ISO 3166 standard [ISO3166]). For example, en-uk for English used in the United Kingdom.
RELATION	VARCHAR(30)	Null	A reference to a related category.
COVERAGE	VARCHAR(30)	Null	The extent or scope of the content of the category.
RIGHTS	VARCHAR(30)	Null	Information about rights held in and over the category.
CREATOR	VARCHAR(50)	Null	An entity primarily responsible for making the content of the category.
PUBLISHER	VARCHAR(50)	Null	An entity responsible for making the category available.
CONTRIBUTOR	VARCHAR(50)	Null	An entity responsible for making contributions to the content of the category.
CREATION_DATE	DATE	Null	A date associated with an event in the life cycle of the category. Recommended best practice for encoding the date value is defined in a profile of ISO 8601 and follows the YYYY-MM-DD format.

Table 8-8 WLCS_CATEGORY Table Metadata (Continued)

Column Name	Data Type	Null Value	Descriptions
MODIFIED_DATE	DATE	Null	A date associated with an event in the life cycle of the category, such as an update or insert by the DBLoader program that is provided with the Commerce Services. The recommended best practice for encoding the date value is defined in a profile of ISO 8601 and follows the YYYY-MM-DD format.
SMALL_IMG_TYPE	NUMBER(3)	Null	A type field of your own design that relates to the graphic. For example, you can implement your own numbering scheme, such as: 0 = display a low resolution graphic for users with low bandwidth. 1 = display a high resolution graphic for users with high bandwidth.
SMALL_IMG_LANG	VARCHAR(30)	Null	The language of the thumbnail image for the category. For related information, see the description of the LANG column.
SMALL_IMG_NAME	VARCHAR(50)	Null	The name of the thumbnail image for the category.
SMALL_IMG_URL	VARCHAR(254)	Null	The URL of the thumbnail image for the category.
SMALL_IMG_ALT_TEXT	VARCHAR(254)	Null	The alternate text to display when users have their cursor over the thumbnail image for the category, or if they have disabled the display of graphics in their browser settings.

Table 8-8 WLCS_CATEGORY Table Metadata (Continued)

Column Name	Data Type	Null Value	Descriptions
LARGE_IMG_TYPE	NUMBER(3)	Null	A type field of your own design that relates to the graphic. For example, you can implement your own numbering scheme, such as: 0 = display a low resolution graphic for users with low bandwidth. 1 = display a high resolution graphic for users with high bandwidth.
LARGE_IMG_LANG	VARCHAR(30)	Null	The language of the full-size image for the category. For related information, see the description of the LANG column.
LARGE_IMG_NAME	VARCHAR(50)	Null	The name of the full-size image for the category.
LARGE_IMG_URL	VARCHAR(254)	Null	The URL of the full-size image for the category.
LARGE_IMG_ALT_TEXT	VARCHAR(254)	Null	The alternate text to display when users have their cursor over the full-size image for the category, or if they have disabled the display of graphics in their browser settings.
DISPLAY_JSP_URL	VARCHAR(254)	Null	The URL to the JSP used to display the category. For example: /commerce/catalog/includes/category.jsp
SHORT_DESC	VARCHAR(50)	Null	A short description of the content of the category.
LONG_DESC	VARCHAR(254)	Null	A long description of the content of the category.

The WLCS_PRODUCT Database Table

This table contains item records in the Commerce database.

Table 8-9 WLCS_PRODUCT Table Metadata

Column Name	Data Type	Null Value	Description
SKU	VARCHAR(40)	Not Null	PK—A unique identifier (the "Stock Keeping Unit," or SKU) for a product item. This field is the table's primary key and cannot be NULL. All other fields in the WLCS_PRODUCT table can be NULL.
IN_STOCK	VARCHAR(1)	Null	A flag to indicate whether the product item is in stock. 0 equates to false, 1 equates to true.
VISIBLE	VARCHAR(1)	Null	Indicates whether the item should be displayed to the user. Enter 1 if visible or 0 if not visible. If not specified in the database, the default is 1.
TAX_CODE	VARCHAR(10)	Null	The code used by the TAXWARE system to identify the specific tax category to which this item belongs.
SHIPPING_CODE	VARCHAR(10)	Null	The code used by the shipping company for this item.
NAME	VARCHAR(100)	Null	A name given to the product item.
SOURCE	VARCHAR(30)	Null	A reference to another product item from which the present item is derived.

Table 8-9 WLCS_PRODUCT Table Metadata (Continued)

Column Name	Data Type	Null Value	Description
LANG	VARCHAR(30)	Null	A language of the intellectual content of the category. The recommended best practice for the values of the language element is defined by RFC 1766, which includes a two-letter Language Code (taken from the ISO 639 standard), such as: en for English; fr for French, or de for German. The language code can, optionally, be followed by a two-letter Country Code (taken from the ISO 3166 standard [ISO3166]). For example, en-uk for English used in the United Kingdom.
RELATION	VARCHAR(30)	Null	A reference to a related product item.
COVERAGE	VARCHAR(30)	Null	The extent or scope of the content of the product item.
RIGHTS	VARCHAR(30)	Null	Information about rights held in and over the item.
FORMAT	VARCHAR(30)	Null	The physical or digital manifestation of the item.
TYPE	VARCHAR(30)	Null	The nature or genre of the content of the item.
MSRP_CURRENCY	VARCHAR(30)	Null	The currency type of the manufacturer's recommended price.
MSRP_AMOUNT	NUMBER(16,4)	Null	The manufacturer's recommended price.
PRICE_CURRENCY	VARCHAR(30)	Null	The currency type of our catalog price for this item.
PRICE_AMOUNT	NUMBER(16,4)	Null	Our current price for this item in the catalog.
ESTIMATE_SHIP_TIME	VARCHAR(100)	Null	Inventory: number of days/weeks before the item can be shipped.
SPECIAL_NOTES	VARCHAR(100)	Null	Inventory related message to display with the item.

Table 8-9 WLCS_PRODUCT Table Metadata (Continued)

Column Name	Data Type	Null Value	Description
CREATOR	VARCHAR(50)	Null	An entity primarily responsible for making the content of the product item.
PUBLISHER	VARCHAR(50)	Null	An entity responsible for making the product item available.
CONTRIBUTOR	VARCHAR(50)	Null	An entity responsible for making contributions to the content of the product item.
CREATION_DATE	DATE	Null	A date associated with an event in the life cycle of the product item. Recommended best practice for encoding the date value is defined in a profile of ISO 8601 and follows the YYYY-MM-DD format.
MODIFIED_DATE	DATE	Null	A date associated with an event in the life cycle of the item, such as an update or insert by the DBLoader program that is provided with the Commerce Services. The recommended best practice for encoding the date value is defined in a profile of ISO 8601 and follows the YYYY-MM-DD format.
SMALL_IMG_TYPE	NUMBER(3)	Null	A type field of your own design that relates to the graphic. For example, you can implement your own numbering scheme, such as: 0 = display a low resolution graphic for users with low bandwidth. 1 = display a high resolution graphic for users with high bandwidth.
SMALL_IMG_LANG	VARCHAR(30)	Null	The language of the thumbnail image for the item. For related information, see the description of the LANG column.
SMALL_IMG_NAME	VARCHAR(50)	Null	The name of the thumbnail image for the item.

Table 8-9 WLCS_PRODUCT Table Metadata (Continued)

Column Name	Data Type	Null Value	Description
SMALL_IMG_URL	VARCHAR(254)	Null	The URL of the thumbnail image for the category.
SMALL_IMG_ALT_TEXT	VARCHAR(254)	Null	The alternate text to display when the user has their cursor over the thumbnail image for the item, or if they have disabled the display of graphics in their browser settings.
LARGE_IMG_TYPE	NUMBER(3)	Null	A type field of your own design that relates to the graphic. For example, you can implement your own numbering scheme, such as: 0 = display a low resolution graphic for users with low bandwidth. 1 = display a high resolution graphic for users with high bandwidth.
LARGE_IMG_LANG	VARCHAR(30)	Null	The language of the full-size image for the item. For related information, see the description of the LANG column.
LARGE_IMG_NAME	VARCHAR(50)	Null	The name of the full-size image for the item.
LARGE_IMG_URL	VARCHAR(254)	Null	The URL of the full-size image for the item.
LARGE_IMG_ALT_TEXT	VARCHAR(254)	Null	The alternate text to display when the user has their cursor over the full-size image of the item, or if they have disabled the display of graphics in their browser settings.
SUM_DISPLAY_JSP_URL	VARCHAR(254)	Null	The URL to the JSP used to display the item in summary form. For example: /commerce/catalog/includes/itemsummary.jsp

Table 8-9 WLCS_PRODUCT Table Metadata (Continued)

Column Name	Data Type	Null Value	Description
DET_DISPLAY_JSP_URL	VARCHAR(254)	Null	The URL to the JSP used to display the item in detailed form. For example: /commerce/catalog/includes/itemdetails.jsp
SHORT_DESC	VARCHAR(254)	Null	A short description of the content of the product item.
LONG_DESC	VARCHAR(2000)	Null	A long description of the content of the product item.

The WLCS_PRODUCT_CATEGORY Database Table

This table contains information that shows which product items are associated with product categories.

Table 8-10 WLCS_PRODUCT_CATEGORY Table Metadata

Column Name	Data Type	Null Value	Description
SKU	VARCHAR(40)	Not Null	PK—A unique identifier (the "Stock Keeping Unit," or SKU) for an item. FK to WLCS_PRODUCT.
CATEGORY_ID	VARCHAR(20)	Not Null	PK—A unique identifier for a category. FK to WLCS_CATEGORY.

The WLCS_PRODUCT_KEYWORD Database Table

This table contains keywords that you associate with each product item. The keywords enable rapid retrieval of item records using the search functions on the Web site's pages or Administration pages.

Table 8-11 WLCS_PRODUCT_KEYWORD Table Metadata

Column Name	Data Type	Null Value	Description
KEYWORD	VARCHAR(30)	Not Null	PK - Contains a keyword that you associate with the product item assigned to the unique SKU. Recommendation—for a given item, select a value from a controlled vocabulary or formal classification scheme implemented in your company.
SKU	VARCHAR(40)	Not Null	PK - A unique identifier (the "Stock Keeping Unit," or SKU) for an item. FK to WLCS_PRODUCT.

Order and Discount Database Objects

Figure 8-4 and Figure 8-5 show the Entity-Relation diagram for the WebLogic Portal order and discount objects.

Figure 8-4 Entity-Relation Diagram for the Commerce Tables

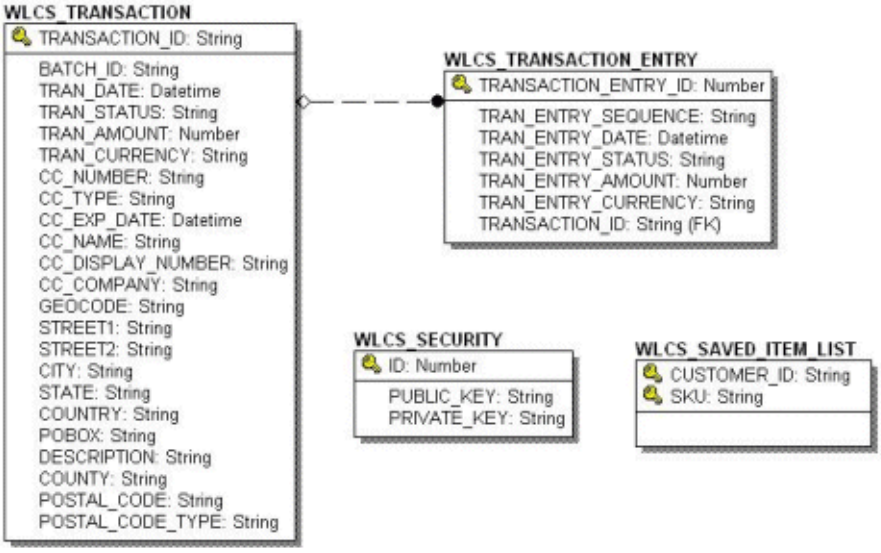
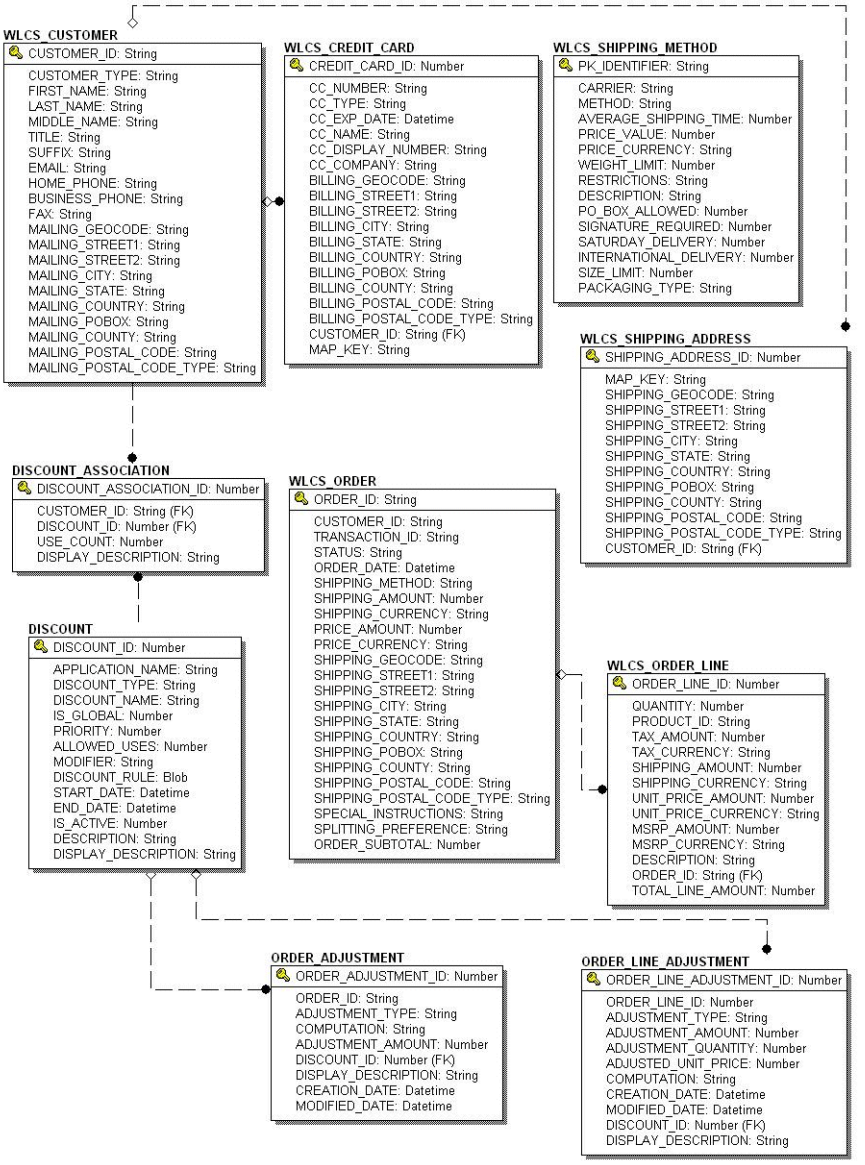


Figure 8-5 Entity-Relation Diagram for the Commerce Tables - continued



The Order Processing Data Dictionary Tables

The Commerce Services order management system has the following tables:

- The DISCOUNT Database Table
- The DISCOUNT_ASSOCIATION Database Table
- The ORDER_ADJUSTMENT Database Table
- The ORDER_LINE_ADJUSTMENT Database Table
- The WLCS_CREDIT_CARD Database Table
- The WLCS_CUSTOMER Database Table
- The WLCS_ORDER Database Table
- The WLCS_ORDER_LINE Database Table
- The WLCS_SAVED_ITEM_LIST Database Table
- The WLCS_SECURITY Database Table
- The WLCS_SHIPPING_ADDRESS Database Table
- The WLCS_SHIPPING_METHOD Database Table
- The WLCS_TRANSACTION Database Table
- The WLCS_TRANSACTION_ENTRY Database Table

The DISCOUNT Database Table

This table contains one or more discount records for every `DISCOUNT_SET` record.

Table 8-12 DISCOUNT

Column Name	Data Type	Null Value	Description
`DISCOUNT_ID`	`NUMBER(15)`	Not Null	PK—A unique, system-generated number to use as the record ID.
`APPLICATION_NAME`	`VARCHAR(100)`	Not Null	FK—Foreign key to the `DISCOUNT_SET` table.

Table 8-12 DISCOUNT (Continued)

Column Name	Data Type	Null Value	Description
DISCOUNT_TYPE	VARCHAR(10)	Not Null	The type of discount offered. It is used for an *order* or for an *order line item*.
DISCOUNT_NAME	VARCHAR(254)	Not Null	The name of the discount.
IS_GLOBAL	NUMBER(1)	Not Null	A flag showing whether or not this discount can be used globally.
PRIORITY	NUMBER(3)	Not Null	The level of priority this discount has over other discounts.
ALLOWED_USERS	NUMBER(10)	Not Null	The number of times the discount can be used.
MODIFIER	VARCHAR(254)	Not Null	Describes the actual discount to be applied. This is XML.
DISCOUNT_RULE	CLOB	Not Null	The method used to select items for discount. This is XML.
START_DATE	DATE	Not Null	The starting date and time of the discount
END_DATE	DATE	Not Null	The ending date and time of the discount.
IS_ACTIVE	NUMBER(1)	Not Null	A flag that determines whether the discount is active or not. Active=1, Not active=0
DESCRIPTION	VARCHAR(254)	Null	The discount description.
DISPLAY_DESCRIPTION	VARCHAR(254)	Null	The discount description used for display purposes only.

The DISCOUNT_ASSOCIATION Database Table

This table contains information that associates each customer with a discount and maintains information regarding the times the customer has used each discount.

Table 8-13 DISCOUNT_ASSOCIATION

Column Name	Data Type	Null Value	Description
DISCOUNT_ASSOCIATION_ID	NUMBER(15)	Not Null	PK—A unique, system-generated number to use as the record ID.
CUSTOMER_ID	VARCHAR(20)	Not Null	FK—Foreign key to the DISCOUNT_SET table.
DISCOUNT_ID	NUMBER(15)	Not Null	FK—Foreign key to the DISCOUNT_SET table.
USE_COUNT	NUMBER(10)	Not Null	The number of times the discount has been used.
DISPLAY_DESCRIPTION	VARCHAR(254)	Null	The discount description used for display purposes only.

The ORDER_ADJUSTMENT Database Table

This table contains information about a discount taken at the order level (for example, $20.00 off any order between 1/1/02 and 1/31/02).

Table 8-14 ORDER_ADJUSTMENT

Column Name	Data Type	Null Value	Description
ORDER_ADJUSTMENT_ID	NUMBER(15)	Not Null	PK—A unique, system-generated number to use as the record ID.
ORDER_ID	VARCHAR(20)	Not Null	FK—Foreign key to the DISCOUNT_SET table.
ADJUSTMENT_TYPE	VARCHAR(20)	Null	The type of adjustment being made to the order line item (for example, order line discount, shipping discount, and so on).
COMPUTATION	VARCHAR(254)	Not Null	The number of times the discount has been used.

Table 8-14 ORDER_ADJUSTMENT (Continued)

Column Name	Data Type	Null Value	Description
ADJUSTMENT_AMOUNT	NUMBER(16,4)	Not Null	The discount description used for display purposes only.
DISCOUNT_ID	NUMBER(15)	Null	FK—Foreign key to the DISCOUNT table.
DISPLAY_DESCRIPTION	VARCHAR(254)	Null	The description used for display purposes only. Depending on the nature of the discount, the DISPLAY_DESCRIPTION is generated from either the Discount service or Campaign service.
CREATION_DATE	DATE	Not Null	The date and time the order adjustment was created.
MODIFIED_DATE	DATE	Null	The date and time the order adjustment record was last modified.

The ORDER_LINE_ADJUSTMENT Database Table

This table contains information about a discount taken at the order line item level (for example, 10% off SKU "Power Drill").

Table 8-15 ORDER_LINE_ADJUSTMENT Table Metadata

Column Name	Data Type	Null Value	Description
ORDER_LINE_ ADJUSTMENT_ID	NUMBER(15)	Not Null	PK—A unique, system-generated number to use as the record ID.
ORDER_LINE_ID	NUMBER(15)	Not Null	A unique identifier for each line in a customer's shopping cart. This field is the table's primary key and cannot be NULL. All other fields in the WLCS_ORDERLINE table can be NULL.
ADJUSTMENT_TYPE	VARCHAR(20)	Null	The type of adjustment being made to the order line item (for example, order line discount, shipping discount, and so on).
ADJUSTMENT_AMOUNT	NUMBER(16,4)	Not Null	The dollar amount of the adjustment.
ADJUSTMENT_QUANTITY	NUMBER(16,4)	Not Null	The quantity amount for the adjustment.

Table 8-15 ORDER_LINE_ADJUSTMENT Table Metadata (Continued)

Column Name	Data Type	Null Value	Description
ADJUSTED_UNIT_PRICE	NUMBER(16,4)	Not Null	The adjusted unit price of the specific line item.
COMPUTATION	VARCHAR(254)	Not Null	The computation for determining ADJUSTED_UNIT_PRICE.
CREATION_DATE	DATE	Not Null	The date and time the adjustment record was created.
MODIFIED_DATE	DATE	Null	The date and time the adjustment record was last modified.
DISCOUNT_ID	NUMBER(15)	Null	FK—a foreign key to the discount used from the DISCOUNT table.
DISPLAY_DESCRIPTION	VARCHAR(254)	Null	The adjustment description used for display purposes.

The WLCS_CREDIT_CARD Database Table

This table contains information related to a customer's credit card(s) in the order processing database.

Table 8-16 WLCS_CREDIT_CARD Table Metadata

Column Name	Data Type	Null Value	Description and Recommendations
CREDIT_CARD_ID	NUMBER(15)	Not Null	A unique identifier for the credit card. This field is the table's primary key and cannot be NULL. All other fields in the WLCS_CREDIT_CARD table can be NULL.
CC_NUMBER	VARCHAR(200)	Null	The customer's credit card number. This is encrypted if is.encryption.enable is set to true in the weblogiccommerce.properties file.
CC_TYPE	VARCHAR(20)	Null	The customer's credit card type, such as VISA or MasterCard.

Table 8-16 WLCS_CREDIT_CARD Table Metadata (Continued)

Column Name	Data Type	Null Value	Description and Recommendations
CC_EXP_DATE	DATE	Null	The expiration date on the customer's credit card.
CC_NAME	VARCHAR(50)	Null	The credit card holder's name.
CC_DISPLAY_NUMBER	VARCHAR(20)	Null	The version of the credit card number that is displayed (all Xs except last 4-digits).
CC_COMPANY	VARCHAR(50)	Null	The name of the credit card company.
BILLING_GEOCODE	VARCHAR(2)	Null	The code used by the TAXWARE system to identify taxes for the order based on jurisdiction.
BILLING_STREET1	VARCHAR(30)	Null	The first line in the customer's billing address.
BILLING_STREET2	VARCHAR(30)	Null	The second line in the customer's billing address.
BILLING_CITY	VARCHAR(30)	Null	The city in the customer's billing address.
BILLING_STATE	VARCHAR(40)	Null	The state in the customer's billing address.
BILLING_COUNTRY	VARCHAR(40)	Null	The country in the customer's billing address.
BILLING_POBOX	VARCHAR(30)	Null	The post office box in the customer's billing address.
BILLING_COUNTY	VARCHAR(50)	Null	The county in the customer's billing address.
BILLING_POSTAL_CODE	VARCHAR(10)	Null	The postal (ZIP) code in the customer's billing address.
BILLING_POSTAL_CODE_ TYPE	VARCHAR(10)	Null	Format or type of postal code, generally determined by country (such as ZIP code in the United States).

Table 8-16 WLCS_CREDIT_CARD Table Metadata (Continued)

Column Name	Data Type	Null Value	Description and Recommendations
CUSTOMER_ID	VARCHAR(20)	Null	A unique identifier for the customer.
MAP_KEY	VARCHAR(60)	Null	Key that maps multiple credit cards with a single customer.

The WLCS_CUSTOMER Database Table

This table contains information about the customer in the order processing database.

Table 8-17 WLCS_CUSTOMER Table Metadata

Column Name	Data Type	Null Value	Description
CUSTOMER_ID	VARCHAR(20)	Not Null	A unique identifier for the customer. This field is the table's primary key and cannot be NULL. All other fields in the WLCS_CUSTOMER table can be NULL.
CUSTOMER_TYPE	VARCHAR(20)	Null	A label for the customer (such as preferred, standard, or business).
FIRST_NAME	VARCHAR(30)	Null	The customer's first name.
LAST_NAME	VARCHAR(30)	Null	The customer's last name.
MIDDLE_NAME	VARCHAR(30)	Null	The customer's middle name.
TITLE	VARCHAR(10)	Null	The customer's preferred title, such as Mr., Mrs., or Ms.
SUFFIX	VARCHAR(10)	Null	The customer's preferred suffix, such as Jr. or Sr.
EMAIL	VARCHAR(80)	Null	The customer's e-mail address.
HOME_PHONE	VARCHAR(15)	Null	The customer's home phone number.
BUSINESS_PHONE	VARCHAR(20)	Null	The customer's business phone number.
FAX	VARCHAR(15)	Null	The customer's fax number.

Table 8-17 WLCS_CUSTOMER Table Metadata (Continued)

Column Name	Data Type	Null Value	Description
MAILING_GEOCODE	VARCHAR(2)	Null	The code used by the TAXWARE system to identify taxes for the order based on jurisdiction.
MAILING_STREET1	VARCHAR(30)	Null	The first line in the customer's street address.
MAILING_STREET2	VARCHAR(30)	Null	The second line in the customer's street address.
MAILING_CITY	VARCHAR(30)	Null	The city in the customer's address.
MAILING_STATE	VARCHAR(40)	Null	The state in the customer's address.
MAILING_COUNTRY	VARCHAR(40)	Null	The country in the customer's address.
MAILING_POBOX	VARCHAR(30)	Null	The post office box in the customer's address.
MAILING_COUNTY	VARCHAR(50)	Null	The county in the customer's address.
MAILING_POSTAL_CODE	VARCHAR(10)	Null	The postal (ZIP) code in the customer's address.
MAILING_POSTAL_CODE_TYPE	VARCHAR(10)	Null	Format or type of postal code, generally determined by country (such as ZIP code in the United States).

The WLCS_ORDER Database Table

This table contains information about a customer's specific order in the order processing database. The Commerce Services product does not populate the SHIPPING_AMOUNT, SHIPPING_CURRENCY, PRICE_AMOUNT, or PRICE_CURRENCY columns.

Table 8-18 WLCS_ORDER Table Metadata

Column Name	Data Type	Null Value	Description
ORDER_ID	VARCHAR(20)	Not Null	PK—A unique identifier for the order. This field is the table's primary key and cannot be NULL. All other fields in the WLCS_ORDER table can be NULL.
CUSTOMER_ID	VARCHAR(20)	Null	A unique identifier for the customer.
TRANSACTION_ID	VARCHAR(25)	Null	A unique identifier for the transaction.
STATUS	VARCHAR(20)	Null	The status of the order.
ORDER_DATE	DATE	Null	The date the order was placed.
SHIPPING_METHOD	VARCHAR(40)	Null	The method by which the order is to be shipped.
SHIPPING_AMOUNT	NUMBER(16,4)	Null	The shipping amount for the order.
SHIPPING_CURRENCY	VARCHAR(10)	Null	The currency associated with the shipping amount.
PRICE_AMOUNT	NUMBER(16,4)	Null	The price of the order.
PRICE_CURRENCY	VARCHAR(10)	Null	The currency associated with the price.
SHIPPING_GEOGODE	VARCHAR(2)	Null	The code used by the TAXWARE system to identify taxes for the order based on jurisdiction.
SHIPPING_STREET1	VARCHAR(30)	Null	The first line in the customer's shipping address.
SHIPPING_STREET2	VARCHAR(30)	Null	The second line in the customer's shipping address.

Table 8-18 WLCS_ORDER Table Metadata (Continued)

Column Name	Data Type	Null Value	Description
SHIPPING_CITY	VARCHAR(30)	Null	The city in the customer's shipping address.
SHIPPING_STATE	VARCHAR(40)	Null	The state in the customer's shipping address.
SHIPPING_COUNTRY	VARCHAR(40)	Null	The country in the customer's shipping address.
SHIPPING_POBOX	VARCHAR(30)	Null	The post office box in the customer's shipping address.
SHIPPING_COUNTY	VARCHAR(50)	Null	The county in the customer's shipping address.
SHIPPING_POSTAL_CODE	VARCHAR(10)	Null	The postal (ZIP) code in the customer's shipping address.
SHIPPING_POSTAL_CODE_TYPE	VARCHAR(10)	Null	Format or type of postal code, generally determined by country, such as ZIP code in the United States.
SPECIAL_INSTRUCTIONS	VARCHAR(254)	Null	Any special shipping instructions associated with the order.
SPLITTING_PREFERENCE	VARCHAR(254)	Null	The splitting preferences for the customer's order.
ORDER_SUBTOTAL	NUMBER(16,4)	Null	The sum of all the TOTAL_LINE_AMOUNT columns in the WLCS_ORDER_LINE table for that specific order.

The WLCS_ORDER_LINE Database Table

This table contains information about each line of a customer's shopping cart in the order processing database.

Table 8-19 WLCS_ORDER_LINE Table Metadata

Column Name	Data Type	Null Value	Description
ORDER_LINE_ID	NUMBER(15)	Not Null	PK—A unique identifier for each line in a customer's shopping cart. This field is the table's primary key and cannot be NULL. All other fields in the WLCS_ORDER_LINE table can be NULL.
QUANTITY	NUMBER(16,4)	Null	The quantity of the item in the shopping cart.
PRODUCT_ID	VARCHAR(40)	Null	An identification number for the item in the shopping cart.
TAX_AMOUNT	NUMBER(16,4)	Null	The tax amount for the order.
TAX_CURRENCY	VARCHAR(10)	Null	The currency associated with the tax amount.
SHIPPING_AMOUNT	NUMBER(16,4)	Null	The shipping amount for the order.
SHIPPING_CURRENCY	VARCHAR(10)	Null	The currency associated with the shipping amount.
UNIT_PRICE_AMOUNT	NUMBER(16,4)	Null	The unit price amount for the item.
UNIT_PRICE_CURRENCY	VARCHAR(10)	Null	The currency associated with the unit price.
MSRP_AMOUNT	NUMBER(16,4)	Null	The MSRP amount for the item.
MSRP_CURRENCY	VARCHAR(10)	Null	The currency associated with the MSRP amount.
DESCRIPTION	VARCHAR(254)	Null	The name of the item that is part of the order.
ORDER_ID	VARCHAR(20)	Null	FK - A unique identifier for the order.

Table 8-19 WLCS_ORDER_LINE Table Metadata (Continued)

Column Name	Data Type	Null Value	Description
TOTAL_LINE_AMOUNT	NUMBER(16,4)	Null	The total discounted price for the line item. UNIT_PRICE_AMOUNT (less any discount) times the QUANTITY.

The WLCS_SAVED_ITEM_LIST Database Table

This table contains information about the customer's saved shopping cart items in the order processing database.

Table 8-20 WLCS_SAVED_ITEM_LIST Table Metadata

Column Name	Data Type	Null Value	Description
CUSTOMER_ID	VARCHAR(20)	Not Null	PK—A unique identifier for the customer.
SKU	VARCHAR(40)	Not Null	PK—A unique identifier (the Stock Keeping Unit or SKU) for a product item.

The WLCS_SECURITY Database Table

This table persists public and private keys for encryption and decryption purposes in the order processing database. This table is meant for internal use by the Commerce Services product.

Table 8-21 WLCS_SECURITY Table Metadata

Column Name	Data Type	Null Value	Description
ID	NUMBER(5)	Not Null	PK—A unique identifier for the key pair.
PUBLIC_KEY	VARCHAR(2000)	Null	The public key to use for encryption/decryption of credit cards.
PRIVATE_KEY	VARCHAR(2000)	Null	The private key to use for encryption/decryption of credit cards.

The WLCS_SHIPPING_ADDRESS Database Table

This table contains information related to a customer's shipping address(es) in the order processing database.

Table 8-22 WLCS_SHIPPING_ADDRESS Table Metadata

Column Name	Data Type	Null Value	Description
SHIPPING_ADDRESS_ID	NUMBER(15)	Not Null	PK - A unique identifier for the shipping address.
MAP_KEY	VARCHAR(60)	Null	Key that maps multiple shipping addresses with a single customer.
SHIPPING_GEOCODE	VARCHAR(2)	Null	The code used by the TAXWARE system to identify taxes for the order based on jurisdiction.
SHIPPING_STREET1	VARCHAR(30)	Null	The first line in the customer's shipping address.
SHIPPING_STREET2	VARCHAR(30)	Null	The second line in the customer's shipping address.
SHIPPING_CITY	VARCHAR(30)	Null	The city in the customer's shipping address.
SHIPPING_STATE	VARCHAR(40)	Null	The state in the customer's shipping address.
SHIPPING_COUNTRY	VARCHAR(40)	Null	The country in the customer's shipping address.
SHIPPING_POBOX	VARCHAR(30)	Null	The post office box in the customer's shipping address.
SHIPPING_COUNTY	VARCHAR(50)	Null	The county in the customer's shipping address.
SHIPPING_POSTAL_CODE	VARCHAR(10)	Null	The postal (ZIP) code in the customer's shipping address.
SHIPPING_POSTAL_CODE_TYPE	VARCHAR(10)	Null	Format or type of postal code, generally determined by country, such as ZIP code in the United States.

Table 8-22 WLCS_SHIPPING_ADDRESS Table Metadata (Continued)

Column Name	Data Type	Null Value	Description
CUSTOMER_ID	VARCHAR(20)	Null	A unique identifier for the customer.

The WLCS_SHIPPING_METHOD Database Table

This table contains information about the shipping method in the order processing database.

Table 8-23 WLCS_SHIPPING_METHOD Table Metadata

Column Name	Data Type	Null Value	Description
PK_IDENTIFIER	VARCHAR(20)	Not Null	PK - A unique identifier for the shipping method.
CARRIER	VARCHAR(40)	Null	The carrier being used to ship the order, such as UPS or FedEx.
METHOD	VARCHAR(40)	Null	The method by which the order is to be shipped, such as Air, 2nd Day Air, or Parcel Post.
AVERAGE_SHIPPING_TIME	NUMBER	Null	The average number of days it will take the order to arrive.
PRICE_VALUE	NUMBER(16,4)	Null	The amount it will cost to ship the order.
PRICE_CURRENCY	VARCHAR(10)	Null	The currency associated with the PRICE_VALUE column, such as dollars, pounds, or lira.
WEIGHT_LIMIT	NUMBER(16,4)	Null	The weight limit for the shipment.
RESTRICTIONS	VARCHAR(254)	Null	Any restrictions associated with the shipment.
DESCRIPTION	VARCHAR(254)	Null	A description of the shipping method, such as FedEx Overnight or Standard.
PO_BOX_ALLOWED	NUMBER	Null	Specifies whether or not the shipment can be left at a post office box.
SIGNATURE_REQUIRED	NUMBER	Null	Specifies whether or not a signature is required upon receipt of the shipment.

Table 8-23 WLCS_SHIPPING_METHOD Table Metadata (Continued)

Column Name	Data Type	Null Value	Description
SATURDAY_DELIVERY	NUMBER	Null	Specifies whether or not the shipment can be delivered on Saturday.
INTERNATIONAL_DELIVERY	NUMBER	Null	Specifies whether or not international delivery is an option.
SIZE_LIMIT	NUMBER(16,4)	Null	The size limit for the shipment.
PACKAGING_TYPE	VARCHAR(50)	Null	The packaging type for the shipment.

The WLCS_TRANSACTION Database Table

This table contains data for every payment transaction in the order processing database.

Table 8-24 WLCS_TRANSACTION Table Metadata

Column Name	Data Type	Null Value	Description
TRANSACTION_ID	VARCHAR(25)	Not Null	PK—A unique identifier for the transaction.
BATCH_ID	VARCHAR(15)	Null	A unique identifier of a batch submitted for settlement, as returned by the Payment Web service. This field need not be populated for other external payment services.
TRAN_DATE	DATE	Null	The date of the transaction (that is, date on which the transaction was first started).
TRAN_STATUS	VARCHAR(20)	Null	The current status of the transaction (Settled, Authorized, MarkedForSettle, PendingSettle, Retry, or Settled).
TRAN_AMOUNT	NUMBER(16,4)	Null	The most recent amount applied to the transaction. MarkForSettle amounts can be different from the authorization amount.
TRAN_CURRENCY	VARCHAR(30)	Null	The currency of the transaction.

Table 8-24 WLCS_TRANSACTION Table Metadata (Continued)

Column Name	Data Type	Null Value	Description
CC_NUMBER	VARCHAR(200)	Null	The customer's credit card number. This is encrypted if `is.encryption.enable` is set to `true` in the `weblogiccommerce.properties` file.
CC_TYPE	VARCHAR(20)	Null	The customer's credit card type, such as VISA or MasterCard.
CC_EXP_DATE	DATE	Null	The expiration date on the customer's credit card.
CC_NAME	VARCHAR(50)	Null	The credit card holder's name.
CC_DISPLAY_NUMBER	VARCHAR(20)	Null	The version of the credit card number that is displayed (displays all Xs except last 4-digits).
CC_COMPANY	VARCHAR(50)	Null	The name of the credit card company.
GEOCODE	VARCHAR(2)	Null	The code used by the TAXWARE system to identify taxes for the order based on jurisdiction.
STREET1	VARCHAR(30)	Null	The first line in the customer's street address.
STREET2	VARCHAR(30)	Null	The second line in the customer's street address.
CITY	VARCHAR(30)	Null	The city in the customer's address.
STATE	VARCHAR(40)	Null	The state in the customer's address.
COUNTRY	VARCHAR(40)	Null	The country in the customer's address.
POBOX	VARCHAR(30)	Null	The post office box in the customer's address.
DESCRIPTION	VARCHAR(30)	Null	Any additional data. Can be NULL.
COUNTY	VARCHAR(50)	Null	The county in the customer's address.

Table 8-24 WLCS_TRANSACTION Table Metadata (Continued)

Column Name	Data Type	Null Value	Description
POSTAL_CODE	VARCHAR(10)	Null	The postal (ZIP) code in the customer's address.
POSTAL_CODE_TYPE	VARCHAR(10)	Null	Format or type of postal code, generally determined by country, such as ZIP code in the United States.

The WLCS_TRANSACTION_ENTRY Database Table

This table logs the different states a payment transaction has passed through in the order processing database.

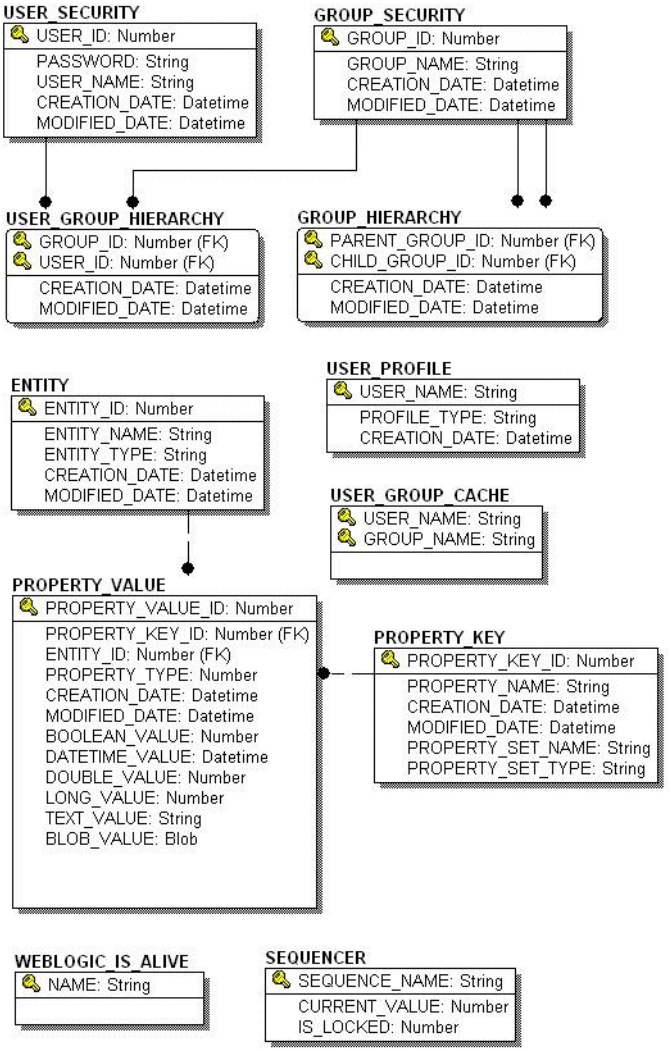
Table 8-25 WLCS_TRANSACTION_ENTRY Table Metadata

Column Name	Data Type	Null Value	Description
TRANSACTION_ENTRY_ID	NUMBER(25)	Not Null	PK— A unique identifier for the transaction entry.
TRAN_ENTRY_SEQUENCE	VARCHAR(30)	Null	Represents the running count per transaction.
TRAN_ENTRY_DATE	DATE	Null	The date of the log entry.
TRAN_ENTRY_STATUS	VARCHAR(20)	Null	The status of the transaction when this entry was made.
TRAN_ENTRY_AMOUNT	NUMBER(16,4)	Null	The amount of the transaction when the log entry was made.
TRAN_ENTRY_CURRENCY	VARCHAR(30)	Null	The currency of the transaction.
TRANSACTION_ID	VARCHAR(25)	Null	A unique identifier for the transaction.

Personalization Database Objects

This section provides information about the database objects for ProductName personalization features. Figure 8-6 shows an entity-relation diagram for the WebLogic Portal Personalization database objects.

Figure 8-6 Entity-Relation Diagram for WebLogic Portal Personalization



The Portal Personalization Database Tables

The following tables compose the portal personalization database:

- The GROUP_HIERARCHY Database Table

- The GROUP_SECURITY Database Table
- The USER_GROUP_CACHE Database Table
- The USER_GROUP_HIERARCHY Database Table
- The USER_PROFILE Database Table
- The USER_SECURITY Database Table
- The ENTITY Database Table
- The PROPERTY_KEY Database Table
- The PROPERTY_VALUE Database Table
- The SEQUENCER Database Table
- The WEBLOGIC_IS_ALIVE Database Table

The GROUP_HIERARCHY Database Table

This table is populated only if the RDBMSAuthenticator is used instead of the default internal LDAP store. This table stores relationship information between groups.

Table 8-26 GROUP_HIERARCHY Table Metadata

Column Name	Data Type	Null Value	Description
PARENT_GROUP_ID	NUMBER(15)	Not Null	PK—The parent group identifier. FK to the ENTITY table.
CHILD_GROUP_ID	NUMBER(15)	Not Null	PK—The child group identifier. FK to the ENTITY table.
CREATION_DATE	DATE	Not Null	The date and time this record was created.
MODIFIED_DATE	DATE	Not Null	The date and time this record was last modified.

The GROUP_SECURITY Database Table

This table is populated only if the RDBMSAuthenticator is used instead of the default internal LDAP store. This table holds all groups that a user could be given membership for security authentication of the RDBMS realm.

Table 8-27 GROUP_SECURITY Table Metadata

Column Name	Data Type	Null Value	Description
GROUP_ID	NUMBER(15)	Not Null	PK—A unique, system-generated number used as the record identifier.
GROUP_NAME	VARCHAR(200)	Not Null	The name of the group.
CREATION_DATE	DATE	Not Null	The date and time this record was created.
MODIFIED_DATE	DATE	Not Null	The date and time this record was last modified.

The USER_GROUP_CACHE Database Table

This table is populated only if the RDBMSAuthenticator is used instead of the default internal LDAP store. In the event of a deep group hierarchy, this table flattens the group hierarchy and enables quick group membership searches.

Note: The startup process GroupCache is disabled by default. This table is used only if enabled.

Table 8-28 USER_GROUP_CACHE Table Metadata

Column Name	Data Type	Null Value	Description
USER_NAME	VARCHAR(200)	Not Null	PK - A user name.
GROUP_NAME	VARCHAR(200)	Not Null	PK - A group name.

The USER_GROUP_HIERARCHY Database Table

This table is populated only if the RDBMSAuthenticator is used instead of the default internal LDAP store. This table allows you to store associated users and groups.

Table 8-29 USER_GROUP_HIERARCHY Table Metadata

Column Name	Data Type	Null Value	Description
GROUP_ID	NUMBER(15)	Not Null	PK - and FK – to USER_SECURITY.USER_ID

Table 8-29 USER_GROUP_HIERARCHY Table Metadata (Continued)

Column Name	Data Type	Null Value	Description
USER_ID	NUMBER(15)	Not Null	PK and FK to GROUP_SECURITY.GROUP_ID
CREATION_DATE	DATE	Not Null	The date and time this record was created.
MODIFIED_DATE	DATE	Not Null	The date and time this record was last modified.

The USER_PROFILE Database Table

This table associates users with profiles (such as the WLCS_CUSTOMER user profile). User profiles use property sets to organize the properties that they contain.

Table 8-30 USER_PROFILE Table Metadata

Column Name	Data Type	Null Value	Description
USER_NAME	VARCHAR(200)	Not Null	PK—The user name.
PROFILE_TYPE	VARCHAR(100)	Not Null	A type of profile associated with the user (such as WLCS_Customer).
CREATION_DATE	DATE	Not Null	The date and time this record was created.

The USER_SECURITY Database Table

This table is populated only if the RDBMSAuthenticator is used instead of the default internal LDAP store. This table holds all the user records for security authentication.

Table 8-31 USER_SECURITY Table Metadata

Column Name	Data Type	Null Value	Description
USER_ID	NUMBER(15)	Not Null	PK—A unique, system-generated number used as the record identifier.
USER_NAME	VARCHAR(200)	Not Null	The user's name.
PASSWORD	VARCHAR(50)	Null	The user's password.
CREATION_DATE	DATE	Not Null	The date and time this record was created.

Table 8-31 USER_SECURITY Table Metadata (Continued)

Column Name	Data Type	Null Value	Description
MODIFIED_DATE	DATE	Not Null	The date and time this record was last modified.

The ENTITY Database Table

Some objects in WebLogic Portal implement a Java interface called ConfigurableEntity. Any ConfigurableEntity within the system has an entry in this table.

Table 8-32 ENTITY Table Metadata

Column Name	Data Type	Null Value	Description
ENTITY_ID	NUMBER(15)	Not Null	PK—A unique, sequence-generated number used as the record identifier.
ENTITY_NAME	VARCHAR(200)	Not Null	The name of the ConfigurableEntity.
ENTITY_TYPE	VARCHAR(100)	Not Null	Defines the type of ConfigurableEntity.
CREATION_DATE	DATE	Not Null	The date and time this record was created.
MODIFIED_DATE	DATE	Not Null	The date and time this record was last modified.

The PROPERTY_KEY Database Table

Any property assigned to a ConfigurableEntity has a unique PROPERTY_ID. The identifier and associated information is stored in the following table.

Table 8-33 PROPERTY_KEY Table Metadata

Column Name	Data Type	Null Value	Description and Recommendations
PROPERTY_KEY_ID	NUMBER(15)	Not Null	PK—A unique, system-generated number used as the record identifier.
PROPERTY_NAME	VARCHAR(100)	Not Null	The property name.
CREATION_DATE	DATE	Not Null	The date and time this record was created.

Table 8-33 PROPERTY_KEY Table Metadata (Continued)

Column Name	Data Type	Null Value	Description and Recommendations
MODIFIED_DATE	DATE	Not Null	The date and time this record was last modified.
PROPERTY_SET_NAME	VARCHAR(100)	Null	The name of the property set.
PROPERTY_SET_TYPE	VARCHAR(100)	Null	The type the property set.

The PROPERTY_VALUE Database Table

This table stores property values for boolean, datetime, float, integer, text, and user-defined properties.

Table 8-34 PROPERTY_VALUE Table Metadata

Column Name	Data Type	Null Value	Description
PROPERTY_VALUE_ID	NUMBER(15)	Not Null	PK—A unique, system-generated number used as the record identifier.
PROPERTY_KEY_ID	NUMBER(15)	Not Null	FK to PROPERTY_KEY.PROPERTY_KEY_ID
ENTITY_ID	NUMBER(15)	Not Null	FK to ENTITY.ENTITY_ID
PROPERTY_TYPE	NUMBER(1)	Not Null	Valid entries are: 0=Boolean, 1=Integer, 2=Float, 3=Text, 4=Date and Time, 5=User-Defined (BLOB).
CREATION_DATE	DATE	Not Null	The date and time this record was created.
MODIFIED_DATE	DATE	Not Null	The date and time this record was last modified.
BOOLEAN_VALUE	NUMBER(1)	Null	The value for each boolean property identifier.
DATETIME_VALUE	DATE	Null	The value for each date and time property identifier.
DOUBLE_VALUE	NUMBER	Null	The value associated with each float property identifier.

Table 8-34 PROPERTY_VALUE Table Metadata (Continued)

Column Name	Data Type	Null Value	Description
LONG_VALUE	NUMBER(20)	Null	The value associated with the integer property.
TEXT_VALUE	VARCHAR(254)	Null	The value associated with the text property.
BLOB_VALUE	BLOB	Null	The value associated with the user-defined property.

The SEQUENCER Database Table

The SEQUENCER table is used to maintain all of the sequence identifiers (for example, property_meta_data_id_sequence, and so on) used in the application.

Table 8-35 SEQUENCER Table Metadata

Column Name	Data Type	Null Value	Description
SEQUENCE_NAME	VARCHAR(50)	Not Null	PK—A unique name used to identify the sequence.
CURRENT_VALUE	NUMBER(15)	Not Null	The current value of the sequence.
IS_LOCKED	NUMBER(1)	Not Null	This flag identifies whether or not the particular SEQUENCE_ID has been locked for update. This column is being used as a generic locking mechanism that can be used for multiple database environments.

The WEBLOGIC_IS_ALIVE Database Table

This table is used by the JDBC connection pools to ensure that the connection to the database is still alive.

Table 8-36 WEBLOGIC_IS_ALIVE Table Metadata

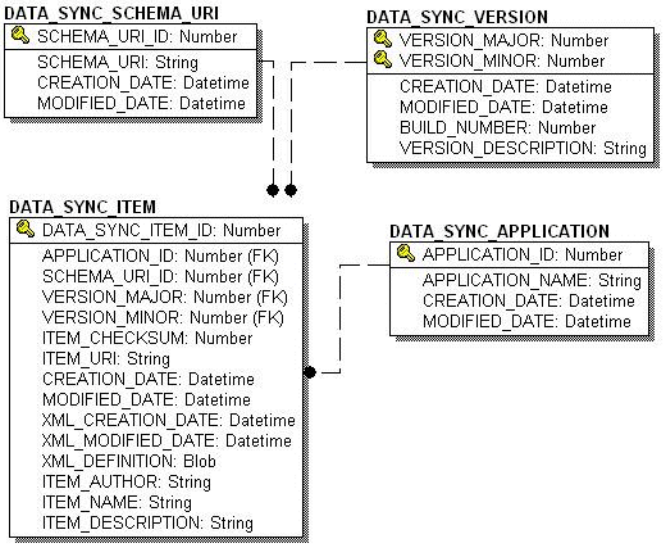
Column Name	Data Type	Null Value	Description
NAME	VARCHAR(100)	Not Null	Used by the JDBC connection pools to ensure that the connection to the database is still alive.

Data Synchronization Database Objects

Data synchronization is a feature that manages XML data about various WebLogic Portal services. Information from the files in the META-INF data folder, under certain circumstances, is written into the data synchronization tables in the database.

This section provides information about the database objects for ProductName data synchronization features. Figure 8-7 shows an entity-relation diagram for WebLogic Portal data synchronization database objects.

Figure 8-7 Entity-Relation Diagram for WebLogic Portal Data Synchronization



The Data Synchronization Database Tables

In this section, WebLogic Portal data synchronization objects tables are arranged alphabetically as a data dictionary.

The following tables compose the data synchronization database:

- The DATA_SYNC_APPLICATION Database Table
- The DATA_SYNC_ITEM Database Table
- The DATA_SYNC_SCHEMA_URI Database Table
- The DATA_SYNC_VERSION Database Table

The DATA_SYNC_APPLICATION Database Table

This table holds the various applications available for the data synchronization process.

Table 8-37 DATA_SYNC_APPLICATION Table Metadata

Column Name	Data Type	Null Value	Description
APPLICATION_ID	NUMBER(15)	Not Null	PK—A unique, system-generated number used as the record identifier.
APPLICATION_NAME	VARCHAR(100)	Not Null	The deployed J2EE application name. This should match the name in the WebLogic Server console.
CREATION_DATE	DATE	Not Null	The date and time this record was created.
MODIFIED_DATE	DATE	Not Null	The date and time this record was last modified.

The DATA_SYNC_ITEM Database Table

This table stores all the data items to be synchronized.

Table 8-38 DATA_SYNC_ITEM Table Metadata

Column Name	Data Type	Null Value	Description
DATA_SYNC_ITEM_ID	NUMBER(15)	Not Null	PK—A unique, system-generated number used as the record identifier.
APPLICATION_ID	NUMBER(15)	Not Null	FK to DATA_SYNC_ APPLICATON.APPLICATION_ID
SCHEMA_URI_ID	NUMBER(15)	Not Null	FK to DATA_SYNC_ SCHEMA_URI.SCHEMA_URI_ID
VERSION_MAJOR	NUMBER(15)	Not Null	FK to DATA_SYNC_ VERSION.VERSION_MAJOR
VERSION_MINOR	NUMBER(15)	Not Null	FK to DATA_SYNC_ VERSION.VERSION_MINOR
ITEM_CHECKSUM	NUMBER(15)	Not Null	A generated number representing the contents of the XML_DEFINITION column.
CREATION_DATE	DATE	Not Null	The date and time this record was created.

Table 8-38 DATA_SYNC_ITEM Table Metadata (Continued)

Column Name	Data Type	Null Value	Description
MODIFIED_DATE	DATE	Not Null	The date and time this record was last modified.
XML_MODIFIED_DATE	DATE	Not Null	The date and time the XML file was last modified.
XML_CREATION_DATE	DATE	Not Null	The date and time the XML file was created.
XML_DEFINITION	CLOB	Not Null	The XML representation of the data item to be synchronized.
ITEM_URI	VARCHAR(254)	Not Null	The path on the file system of the data item to be synchronized.
ITEM_AUTHOR	VARCHAR(200)	Null	Metadata info—the OS login.
ITEM_NAME	VARCHAR(100)	Null	Metadata info—the full path to the item.
ITEM_DESCRIPTION	VARCHAR(254)	Null	Metadata info—a general description of the item to be synchronized.

The DATA_SYNC_SCHEMA_URI Database Table

This table holds information pertaining to each of the governing schemas used by various documents.

Table 8-39 DATA_SYNC_SCHEMA_URI Table Metadata

Column Name	Data Type	Null Value	Description
SCHEMA_URI_ID	NUMBER(15)	Not Null	PK— A unique, system-generated number used as the record identifier.
SCHEMA_URI	VARCHAR(254)	Not Null	The governing schema of the document.
CREATION_DATE	DATE	Not Null	The date and time this record was created.
MODIFIED_DATE	DATE	Not Null	The date and time this record was last modified.

The DATA_SYNC_VERSION Database Table

This table is not being used currently. It is reserved for future use and is expected to accommodate data synchronization versioning. As a result, this table holds only one record.

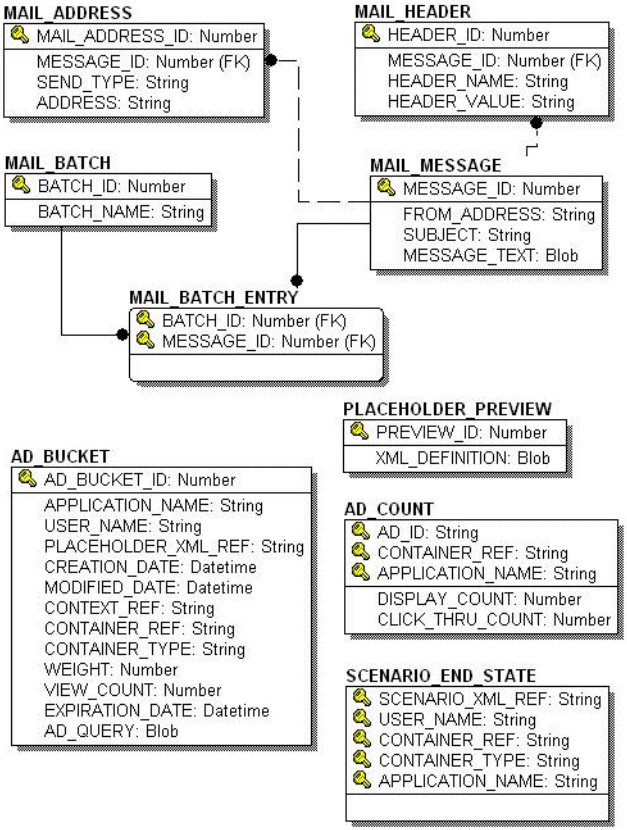
Table 8-40 DATA_SYNC_VERSION Table Metadata

Column Name	Data Type	Null Value	Description
VERSION_MAJOR	NUMBER(15)	Not Null	PK—The current record has a value of zero.
VERSION_MINOR	NUMBER(15)	Not Null	PK—The current record has a value of zero.
CREATION_DATE	DATE	Not Null	The date and time that the record was created.
MODIFIED_DATE	DATE	Not Null	The date and time that the record was last modified.
BUILD_NUMBER	NUMBER(15)	Null	The build number associated with the version.
VERSION_DESCRIPTION	VARCHAR(30)	Null	A description of the particular sync version.

WebLogic Portal Services Database Objects

This section provides information about the database objects for ProductName Services features. Figure 8-8 shows an entity-relation diagram for WebLogic Portal services database objects.

Figure 8-8 Entity-Relation Diagram for WebLogic Portal Services



The Portal Services Database Tables

The following tables compose the Portal services database:

- The AD_BUCKET Database Table
- The AD_COUNT Database Table
- The PLACEHOLDER_PREVIEW Database Table
- The MAIL_ADDRESS Database Table
- The MAIL_BATCH Database Table
- The MAIL_BATCH_ENTRY Database Table
- The MAIL_HEADER Database Table
- The MAIL_MESSAGE Database Table
- The SCENARIO_END_STATE Database Table

The AD_BUCKET Database Table

This table maintains content queries for ads.

Table 8-41 AD_BUCKET Table Metadata

Column Name	Data Type	Null Value	Description
AD_BUCKET_ID	NUMBER(15)	Not Null	PK—A unique, system-generated number used as the record identifier.
USER_NAME	VARCHAR (200)	Not Null	The user's name associated with the ad.
PLACEHOLDER_XML_REF	VARCHAR(254)	Not Null	The location identifier of the XML-based placeholder definition file.
APPLICATION_NAME	VARCHAR(100)	Not Null	The name of the application for which the ad has been scoped.
CONTEXT_REF	VARCHAR(254)	Null	The scenario-unique identifier.
CONTAINER_REF	VARCHAR(254)	Null	The campaign-unique identifier.
CONTAINER_TYPE	VARCHAR(50)	Null	Identifies the service associated with the CONTAINER_REF.

Table 8-41 AD_BUCKET Table Metadata

Column Name	Data Type	Null Value	Description
WEIGHT	NUMBER(15)	Null	A weighted scheme used in prioritizing one placeholder over another.
VIEW_COUNT	NUMBER(15)	Null	*Disabled. Reserved for future use.*
EXPIRATION_DATE	DATE	Null	The date and time the ad expires or becomes invalid.
CREATION_DATE	DATE	Not Null	The date and time this record was created.
MODIFIED_DATE	DATE	Not Null	The date and time this record was last modified.
AD_QUERY	CLOB	Null	The actual content query.

The AD_COUNT Database Table

This table tracks the number of times the ads are displayed and clicked though.

Table 8-42 AD_COUNT Table Metadata

Column Name	Data Type	Null Value	Description
AD_ID	VARCHAR(254)	Not Null	PK—A unique, system-generated number used as the record identifier.
CONTAINER_REF	VARCHAR(254)	Not Null	PK—The campaign-unique identifier.
APPLICATION_NAME	VARCHAR(100)	Not Null	PK—The name of the application for which the ad clicks or views were scoped.
DISPLAY_COUNT	NUMBER(15)	Not Null	The number of times the ad has been displayed.
CLICK_THROUGH_COUNT	NUMBER(15)	Not Null	The number of times the ad has been clicked.

The PLACEHOLDER_PREVIEW Database Table

This table is used as a mechanism to hold the placeholder for previewing purposes only.

Table 8-43 PLACEHOLDER_PREVIEW Table Metadata

Column Name	Data Type	Null Value	Description
PREVIEW_ID	NUMBER	Not Null	PK—A unique, system-generated number used as the record identifier.
XML_DEFINITION	CLOB	Null	The representation of the expression to be previewed.

The MAIL_ADDRESS Database Table

This table stores all of the address information for e-mail purposes.

Table 8-44 MAIL_ADDRESS Table Metadata

Column Name	Data Type	Null Value	Description
MAIL_ADDRESS_ID	NUMBER(15)	Not Null	PK—A unique, system-generated number to use as the record ID.
MESSAGE_ID	NUMBER(15)	Not Null	FK—Foreign key to the MAIL_MESSAGE table.
ADDRESS	VARCHAR(254)	Not Null	Stores the various e-mail addresses on the distribution list.
SEND_TYPE	VARCHAR(4)	Not Null	Determines how the ADDRESS should be included on the distribution. Possible values are TO, CC, or BCC.

The MAIL_BATCH Database Table

This table establishes a batch for each mailing.

Table 8-45 MAIL_BATCH Table Metadata

Column Name	Data Type	Null Value	Description
BATCH_ID	NUMBER(15)	Not Null	PK—A unique, system-generated number to use as the record ID.

Table 8-45 MAIL_BATCH Table Metadata (Continued)

Column Name	Data Type	Null Value	Description
BATCH_NAME	VARCHAR(254)	Not Null	The name of the mail message batch.

The MAIL_BATCH_ENTRY Database Table

This table is used to correlate the mail batch with the specific mail message.

Table 8-46 MAIL_BATCH_ENTRY Table Metadata

Column Name	Data Type	Null Value	Description
BATCH_ID	NUMBER(15)	Not Null	PK and FK—A unique, system-generated number to use as the record ID.
MESSAGE_ID	NUMBER(15)	Not Null	PK and FK—Foreign key to the MAIL_MESSAGE table.

The MAIL_HEADER Database Table

This table contains all of the header information specific to the e-mail message.

Table 8-47 MAIL_HEADER Table Metadata

Column Name	Data Type	Null Value	Description
HEADER_ID	NUMBER(15)	Not Null	PK—A unique, system-generated number to use as the record ID.
MESSAGE_ID	NUMBER(15)	Not Null	FK—Foreign key to the MAIL_MESSAGE table.
HEADER_NAME	VARCHAR(50)	Null	The name of the mail message header.
HEADER_VALUE	VARCHAR(254)	Null	The value of the mail message header.

The MAIL_MESSAGE Database Table

This table contains the specifics of the mail message (for example, the subject line, text, and so on).

Table 8-48 MAIL_MESSAGE Table Metadata

Column Name	Data Type	Null Value	Description
MESSAGE_ID	NUMBER(15)	Not Null	PK—A unique, system-generated number to use as the record ID.
FROM_ADDRESS	VARCHAR(254)	Null	Identifies who is sending the message.
SUBJECT	VARCHAR(128)	Null	Stores the mail message subject.
MESSAGE_TEXT	CLOB	Null	Holds the content of the mail message.

The SCENARIO_END_STATE Database Table

This table identifies when a user is no longer eligible to participate in a particular scenario.

Table 8-49 SCENARIO_END_STATE Table Metadata

Column Name	Data Type	Null Value	Description
SCENARIO_XML_REF	VARCHAR(20)	Not Null	PK—The identifier for the XML-based scenario definition file.
USER_NAME	VARCHAR(200)	Not Null	PK—the user ID. FK to WLCS_USER.IDENTIFIER.
CONTAINER_REF	VARCHAR(254)	Not Null	PK—the campaign unique identifier. FK to CAMPAIGN.CAMPAIGN_UID.
CONTAINER_TYPE	VARCHAR(50)	Not Null	PK—At this time this column always holds the string Campaign.
APPLICATION_NAME	VARCHAR(100)	Not Null	PK—The deployed J2EE application name. This should match the name in the WebLogic Server console.

Portal Framework Database Objects

This section documents the database objects for the WebLogic Portal package. Figure 8-9 and xxx (in two pages) show the entity-relation diagram for the WebLogic Portal Framework database objects.

Figure 8-9 Entity-Relation Diagram for the Portal Framework Tables (page 1 of 2)

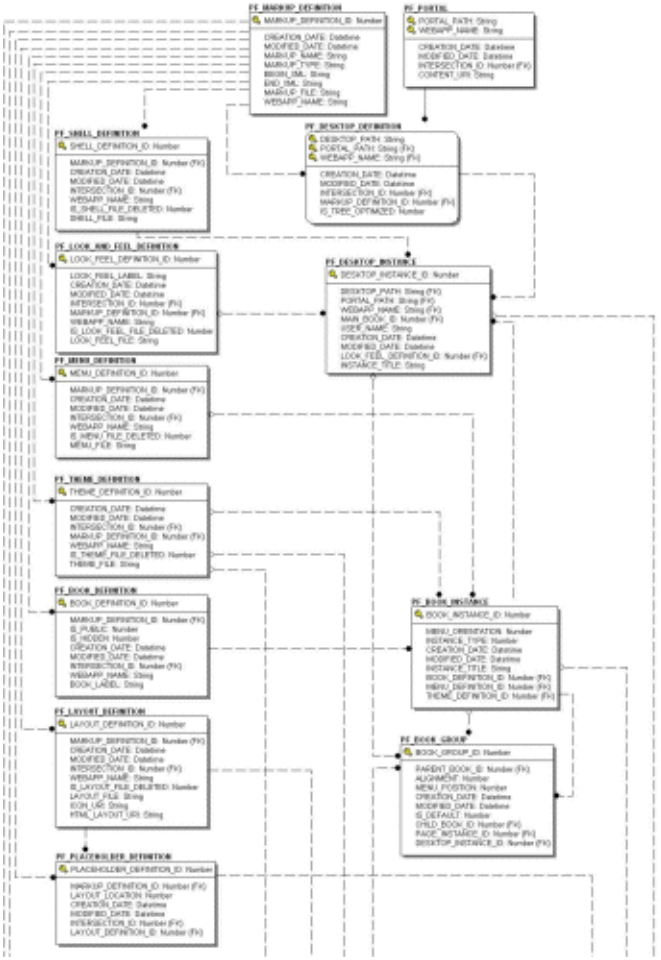
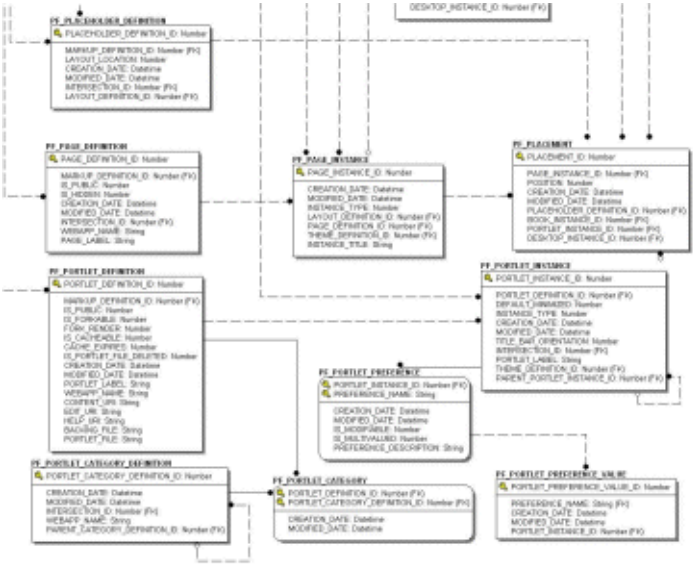


Figure 8-10 Entity-Relation Diagram for the Portal Framework Tables (page 2 of 2)



The Portal Framework Database Tables

In this section, WebLogic Portal Services tables are arranged alphabetically as a data dictionary. The following tables compose the Portal Framework database:

- The PF_BOOK_DEFINITION Database Table
- The PF_BOOK_GROUP Database Table
- The PF_BOOK_INSTANCE Database Table
- The PF_DESKTOP_DEFINITION Database Table
- The PF_DESKTOP_INSTANCE Database Table
- The PF_LAYOUT_DEFINITION Database Table
- The PF_LOOK_AND_FEEL_DEFINITION Database Table
- The PF_MARKUP_DEFINITION Database Table
- The PF_MENU_DEFINITION Database Table
- The PF_PAGE_DEFINITION Database Table

- The PF_PAGE_INSTANCE Database Table
- The PF_PLACEHOLDER_DEFINITION Database Table
- The PF_PLACEMENT Database Table
- The PF_PORTAL Database Table
- The PF_PORTLET_CATEGORY Database Table
- The PF_PORTLET_CATEGORY_DEFINITION Database Table
- The PF_PORTLET_DEFINITION Database Table
- The PF_PORTLET_INSTANCE Database Table

  The following WSRP-related tables are a subset of the PF_PORTLET_INSTANCE database table:

  – The PF_CONSUMER_PORTLETS Database Table
  – The PF_CONSUMER_PROPERTIES Database Table
  – The PF_CONSUMER_REGISTRY Database Table
  – The PF_PRODUCER_PROPERTIES Database Table
  – The PF_PRODUCER_REGISTRY Database Table
  – The PF_PROXY_PORTLET_INSTANCE Database Table

  To view the diagram for the tables listed above, see "Entity-Relation Diagram for the WSRP Tables" on page 8-83.

- The PF_PORTLET_PREFERENCE Database Table
- The PF_PORTLET_PREFERENCE_VALUE Database Table
- The PF_SHELL_DEFINITION Database Table
- The PF_THEME_DEFINITION Database Table

The PF_BOOK_DEFINITION Database Table

This table defines a BOOK portal library resource. Books are used to aggregate PAGES and other BOOKS.

Table 8-50 PF_BOOK_DEFINITION Table Metadata

Column Name	Data Type	Null Value	Description
BOOK_DEFINITION_ID	NUMBER	Not Null	PK—A unique, system-generated number to use as the record ID.
MARKUP_DEFINITION_ID	NUMBER	Not Null	FK to PF_MARKUP_DEFINITION.
IS_PUBLIC	NUMBER	Not Null	A boolean flag indicating whether this book definition displays to the public. When end users create books they are not marked as public.
IS_HIDDEN	NUMBER	Not Null	A boolean flag indicating whether this book definition is hidden from the menu. Marking a page or book as hidden does not prevent it from being displayed; this indicator is only a hint to the menu control not to display a tab for the given book or page. The page or book can be activated using a link or a backing file.
CREATION_DATE	DATE	Not Null	The date and time the row was created.
MODIFIED_DATE	DATE	Not Null	The date and time the row was last modified. This column's data is maintained using a database trigger.
INTERSECTION_ID	NUMBER	Not Null	FK to L10N_INTERSECTION.
WEBAPP_NAME	VARCHAR(80)	Not Null	Name of the J2EE Web application (as defined in the config.xml) to which the portal resource is scoped.
BOOK_LABEL	VARCHAR(80)	Null	A moniker used to reference this portal resource for development purposes. This is the same as the bookDefinitionLabel in WebLogic Workshop. If a label is not supplied at creation time, the BOOK_DEFINITION_ID prefixed with a 'B' is used. This label can be supplied to APIs to activate books or pages.

The PF_BOOK_GROUP Database Table

This table represents a child page or book placement on the parent book. A single record in the table represents one placement on a book. This table also identifies a customized grouping of Books and Pages. Customized groupings are represented and aggregated around the DESKTOP_INSTANCE_ID.

Table 8-51 PF_BOOK_GROUP Table Metadata

Column Name	Data Type	Null Value	Description
BOOK_GROUP_ID	NUMBER	Not Null	PK—A unique, system-generated number to use as the record ID.
PARENT_BOOK_ID	NUMBER	Not Null	FK to PF_BOOK_INSTANCE that identifies the parent BOOK_INSTANCE_ID.
ALIGNMENT	NUMBER	Not Null	The alignment is a 'hint' to the menu skeleton JSP to indicate whether the tab should be aligned on the left or right of the tab bar. A skeleton can either implement this feature or ignore it.
MENU_POSITION	NUMBER	Not Null	The order, in the tab menu, in which this page or book will appear on the parent book.The order does not need to be contiguous.
CREATION_DATE	DATE	Not Null	The date and time the row was created.
MODIFIED_DATE	DATE	Not Null	The date and time the row was last modified. This column's data is maintained using a database trigger.
IS_DEFAULT	NUMBER	Not Null	A boolean flag indicating that this is the default page or book on the parent book.
CHILD_BOOK_ID	NUMBER	Null	FK to PF_BOOK_INSTANCE that identifies the child BOOK_INSTANCE_ID.
PAGE_INSTANCE_ID	NUMBER	Null	FK to PF_BOOK_INSTANCE.

Table 8-51 PF_BOOK_GROUP Table Metadata (Continued)

Column Name	Data Type	Null Value	Description
DESKTOP_INSTANCE_ID	NUMBER	Null	FK to PF_DESKTOP_INSTANCE. If this book grouping is an administrator or end user customization, this value is non null and points to the administrator's or user's desktop. If this field is null, it represents the library's view.

The PF_BOOK_INSTANCE Database Table

This table identifies an instance of the BOOK_DEFINITION. There is always at least one book instance, namely the primary instance. All other instances represent customization by administrators or end users.

Table 8-52 PF_BOOK_INSTANCE Table Metadata

Column Name	Data Type	Null Value	Description
BOOK_INSTANCE_ID	NUMBER	Not Null	PK—A unique, system-generated number to use as the record ID.
MENU_ORIENTATION	NUMBER	Not Null	The orientation is a hint to the book skeleton JSP and the menu skeleton JSP to display the tabs on the top, left, right, or bottom of the main book. The skeletons can choose to ignore this field.
INSTANCE_TYPE	NUMBER	Not Null	The type of book instance: 1=Primary, 3=Admin, 4=User.
CREATION_DATE	DATE	Not Null	The date and time the row was created.
MODIFIED_DATE	DATE	Not Null	The date and time the row was last modified. This column's data is maintained using a database trigger.

Table 8-52 PF_BOOK_INSTANCE Table Metadata (Continued)

Column Name	Data Type	Null Value	Description
INSTANCE_TITLE	VARCHAR(255)	Null	An end-user-customized title for this BOOK. This title is not internationalized as it is used only by the end user. If the end user does not customize the book title, then the value is null and the L10N_RESOURCE title is used.
BOOK_DEFINITION_ID	NUMBER	Not Null	FK to PF_BOOK_DEFINITION.
MENU_DEFINITION_ID	NUMBER	Null	FK to PF_MENU_DEFINITION. Can be null as not every book must have a menu.
THEME_DEFINITION_ID	NUMBER	Null	FK to PF_THEME_DEFINITION.

The PF_DESKTOP_DEFINITION Database Table

This table defines a desktop definition. You can create Desktops from template (.portal) files or from existing resources.

Table 8-53 PF_DESKTOP_DEFINITION Table Metadata

Column Name	Data Type	Null Value	Description
DESKTOP_PATH	VARCHAR(40)	Not Null	Part of the PK— identifies the partial URL path to the desktop.
PORTAL_PATH	VARCHAR(40)	Not Null	Part of the PK and FK to PF_PORTAL— identifies the partial URL path to this desktop and parent portal.
WEBAPP_NAME	VARCHAR(80)	Not Null	Part of the PK and FK to PF_PORTAL. This is the name of the webapp (as defined in the config.xml file) to which this desktop is scoped.
CREATION_DATE	DATE	Not Null	The date and time the row was created.
MODIFIED_DATE	DATE	Not Null	The date and time the row was last modified. This column's data is maintained using a database trigger.

Table 8-53 PF_DESKTOP_DEFINITION Table Metadata (Continued)

Column Name	Data Type	Null Value	Description
INTERSECTION_ID	NUMBER	Not Null	FK to L10N_INTERSECTION. The BOOK_INSTANCE_ID of the main or default PF_BOOK_INSTANCE for the desktop.
MARKUP_DEFINITION_ID	NUMBER	Not Null	FK to PF_MARKUP_DEFINITION.
IS_TREE_OPTIMIZED	NUMBER	Not Null	Indicates whether tree optimization is active for a desktop. Acceptable values are 0 (off; the default) or 1 (on).

The PF_DESKTOP_INSTANCE Database Table

This table identifies a customized or localized instance of a desktop.

Table 8-54 PF_DESKTOP_INSTANCE Table Metadata

Column Name	Data Type	Null Value	Description
DESKTOP_INSTANCE_ID	NUMBER	Not Null	PK—identifies the partial URL path to the desktop.
DESKTOP_PATH	VARCHAR(40	Not Null	FK to PF_DESKTOP_DEFINITION.
PORTAL_PATH	VARCHAR(40)	Not Null	FK to PF_DESKTOP_DEFINITION.
WEBAPP_NAME	VARCHAR(80)	Not Null	FK to PF_DESKTOP_DEFINITION.
MAIN_BOOK_ID	NUMBER	Not Null	FK to BOOK_INSTANCE_ID of the main or default PF_BOOK_INSTANCE for the desktop.
USER_NAME	VARCHAR(200)	Null	The name of the user if the user has customized his/her desktop. This value is null if the desktop instance is not for a particular user or administrator.
CREATION_DATE	DATE	Not Null	The date and time the row was created.

Table 8-54 PF_DESKTOP_INSTANCE Table Metadata (Continued)

Column Name	Data Type	Null Value	Description
MODIFIED_DATE	DATE	Not Null	The date and time the row was last modified. This column's data is maintained using a database trigger.
LOOK_FEEL_DEFINITION_ID	NUMBER	Null	FK to PF_LOOK_AND_FEEL_DEFINITION.
INSTANCE_TITLE	VARCHAR(20)	Null	An end-user-customized title for this DESKTOP. This title is not internationalized as it is used only by the end user. If the end user does not customize the desktop title, then the value is null and the L10N_RESOURCE title is used.
SHELL_DEFINITION_ID	NUMBER	Not Null	FK to PF_SHELL_DEFINITION.

The PF_LAYOUT_DEFINITION Database Table

This table defines a LAYOUT portal library resource that is used as a specification for determining the location of items on a page. For every layout definition there is a corresponding.layout file. By updating the .layout file, you are updating this record.

Table 8-55 PF_LAYOUT_DEFINITION Table Metadata

Column Name	Data Type	Null Value	Description
LAYOUT_DEFINITION_ID	NUMBER	Not Null	PK—A unique, system-generated number to use as the record ID.
MARKUP_DEFINITION_ID	NUMBER	Not Null	FK to PF_MARKUP_DEFINITION.
CREATION_DATE	DATE	Not Null	The date and time the row was created.
MODIFIED_DATE	DATE	Not Null	The date and time the row was last modified. This column's data is maintained using a database trigger.
INTERSECTION_ID	NUMBER	Not Null	FK to L10N_INTERSECTION.

Table 8-55 PF_LAYOUT_DEFINITION Table Metadata (Continued)

Column Name	Data Type	Null Value	Description
WEBAPP_NAME	VARCHAR(80)	Not Null	Name of the J2EE Web application to which the portal resource is scoped.
IS_LAYOUT_FILE_ DELETED	NUMBER	Not Null	A boolean indicating that the file associated with this layout was removed from the file system. If the layout is not being used, then the record is deleted outright. This flag is set to true only when the .layout file is deleted and the layout is still in use. You can either return the .layout file and this flag is automatically reset, or remove the layout with a replacement layout in the admin tools.
LAYOUT_FILE	VARCHAR(255)	Null	The name and location of the file associated with this layout definition.
ICON_URI	VARCHAR(255)	Null	The URI that identifies the ICON for this layout definition.
HTML_LAYOUT_URI	VARCHAR(255)	Null	The URI for the HTML for this layout definition. The htlp file is used by the admin and visitor tools to provide a visual display that emulates the real layout.

The PF_LOOK_AND_FEEL_DEFINITION Database Table

This table defines a LOOK and FEEL portal library resource or template for assignment to DESKTOPs that control how a portal renders.

Table 8-56 PF_LOOK_AND_FEEL_DEFINITION Table Metadata

Column Name	Data Type	Null Value	Description
LOOK_FEEL_ DEFINITION_ID	NUMBER	Not Null	PK—A unique, system-generated number to use as the record ID.
LOOK_FEEL_LABEL	VARCHAR(80)	Not Null	A moniker used to reference this portal resource for development purposes.

Table 8-56 PF_LOOK_AND_FEEL_DEFINITION Table Metadata (Continued)

Column Name	Data Type	Null Value	Description
CREATION_DATE	DATE	Not Null	The date and time the row was created.
MODIFIED_DATE	DATE	Not Null	The date and time the row was last modified. This column's data is maintained using a database trigger.
INTERSECTION_ID	NUMBER	Not Null	FK to L10N_INTERSECTION.
MARKUP_DEFINITION_ID	NUMBER	Not Null	FK to PF_MARKUP_DEFINITION.
WEBAPP_NAME	VARCHAR(80)	Not Null	Name of the J2EE Web application to which the portal resource is scoped.
IS_LOOK_FEEL_FILE_ DELETED	NUMBER	Not Null	A boolean indicating that the file associated with this look and feel was removed from the file system. If the look and feel is not being used, then the record is deleted outright. This flag is set to true only when the `.laf` file is deleted and the look and feel is still in use. You can either return the `.laf` file and this flag is automatically reset, or remove the look and feel with a replacement look and feel in the WebLogic Administration Portal.
LOOK_FEEL_FILE	VARCHAR(255)	Not Null	The fully qualified file path (from the web app) to the location of the .laf file associated with this look and feel definition.

The PF_MARKUP_DEFINITION Database Table

This table defines the MARKUP (blueprint, design, model) for a portal library resource.

Table 8-57 PF_MARKUP_DEFINITION Table Metadata

Column Name	Data Type	Null Value	Description
MARKUP_DEFINITION_ID	NUMBER	Not Null	PK—A unique, system-generated number to use as the record ID.

Table 8-57 PF_MARKUP_DEFINITION Table Metadata (Continued)

Column Name	Data Type	Null Value	Description
CREATION_DATE	DATE	Not Null	The date and time the row was created.
MODIFIED_DATE	DATE	Not Null	The date and time the row was last modified. This column's data is maintained using a database trigger.
MARKUP_NAME	VARCHAR(255)	Not Null	The file name and location that contains the definition of this portal object.
MARKUP_TYPE	VARCHAR(20)	Not Null	The type of portal resource that this markup defines.
BEGIN_XML	VARCHAR(2000)	Not Null	The first 2000 characters of XML definition of this portal object.
END_XML	VARCHAR(2000)	Null	The last 2000 characters of the XML definition of this portal object.
MARKUP_FILE	VARCHAR(255)	Null	Location of the file containing the markup definition.
WEBAPP_NAME	VARCHAR(80)	Null	Name of the J2EE Web application to which the portal resource is scoped.

The PF_MENU_DEFINITION Database Table

This table defines a MENU portal library resource or template that can be assigned to a BOOK INSTANCE.

Table 8-58 PF_MENU_DEFINITION Table Metadata

Column Name	Data Type	Null Value	Description
MENU_DEFINITION_ID	NUMBER	Not Null	PK—A unique, system-generated number to use as the record ID.
MARKUP_DEFINITION_ID	NUMBER	Not Null	FK to PF_MARKUP_DEFINITION.
CREATION_DATE	DATE	Not Null	The date and time the row was created.

Table 8-58 PF_MENU_DEFINITION Table Metadata (Continued)

Column Name	Data Type	Null Value	Description
MODIFIED_DATE	DATE	Not Null	The date and time the row was last modified. This column's data is maintained using a database trigger.
INTERSECTION_ID	NUMBER	Not Null	FK to L10N_INTERSECTION.
WEBAPP_NAME	VARCHAR(80)	Not Null	Name of the J2EE Web application to which the portal resource is scoped.
IS_MENU_FILE_DELETED	NUMBER	Not Null	A boolean indicating that the file associated with this menu was removed from the file system. If the menu is not being used then the record is deleted outright. This flag is set to true only when the .menu file is deleted and the menu is still in use. You can either return the .menu file and this flag is automatically reset, or remove the menu with a replacement menu in the WebLogic Administration Portal.
MENU_FILE	VARCHAR(255)	Not Null	The fully qualified path (from the Web application) to the location of the .menu file associated with this menu definition.

The PF_PAGE_DEFINITION Database Table

This table defines a PAGE portal library resource or template that can be assigned to a PAGE INSTANCE.

Table 8-59 PF_PAGE_DEFINITION Table Metadata

Column Name	Data Type	Null Value	Description
PAGE_DEFINITION_ID	NUMBER	Not Null	PK—A unique, system-generated number to use as the record ID.
MARKUP_DEFINITION_ID	NUMBER	Not Null	FK to PF_MARKUP_DEFINITION.

Table 8-59 PF_PAGE_DEFINITION Table Metadata (Continued)

Column Name	Data Type	Null Value	Description
IS_PUBLIC	NUMBER	Not Null	A boolean indicating this page definition is public. Only public page definitions are ever exposed to "visitors."
IS_HIDDEN	NUMBER	Not Null	A boolean indicating this page is hidden. The hidden flag is a hint to the menu not to render a tab for this page. The page can still be displayed by other methods (links, events).
CREATION_DATE	DATE	Not Null	The date and time the row was created.
MODIFIED_DATE	DATE	Not Null	The date and time the row was last modified. This column's data is maintained using a database trigger.
INTERSECTION_ID	NUMBER	Not Null	FK to L10N_INTERSECTION.
WEBAPP_NAME	VARCHAR(80)	Not Null	Name of the J2EE Web application to which the portal resource is scoped.
PAGE_LABEL	VARCHAR(80)	Null	A moniker used to reference this portal resource for development purposes.

The PF_PAGE_INSTANCE Database Table

This table identifies an instance of the page definition; at least one instance per definition always exists.

Table 8-60 PF_PAGE_INSTANCE Table Metadata

Column Name	Data Type	Null Value	Description
PAGE_INSTANCE_ID	NUMBER	Not Null	PK—A unique, system-generated number to use as the record ID.
CREATION_DATE	DATE	Not Null	The date and time the row was created.
MODIFIED_DATE	DATE	Not Null	The date and time the row was last modified. This column's data is maintained using a database trigger.

Table 8-60 PF_PAGE_INSTANCE Table Metadata (Continued)

Column Name	Data Type	Null Value	Description
INSTANCE_TYPE	NUMBER	Not Null	The type of page instance: 1=Primary, 3=Admin, 4=User.
LAYOUT_DEFINITION_ID	NUMBER	Not Null	FK to PF_LAYOUT_DEFINITION.
PAGE_DEFINITION_ID	NUMBER	Not Null	FK to PF_PAGE_DEFINITION.
THEME_DEFINITION_ID	NUMBER	Null	FK to PF_THEME_DEFINITION.
INSTANCE_TITLE	VARCHAR(255)	Null	A desktop- or user-customized title for this page. This instance title is valid only to end users as it cannot and need not be localized.

The PF_PLACEHOLDER_DEFINITION Database Table

This table defines a PLACEHOLDER portal library resource or template that has a LAYOUT definition and can be assigned to a PLACEMENT.

Table 8-61 PF_PLACEHOLDER_DEFINITION Table Metadata

Column Name	Data Type	Null Value	Description
PLACEHOLDER_DEFINITION_ID	NUMBER	Not Null	PK—A unique, system-generated number to use as the record ID.
MARKUP_DEFINITION_ID	NUMBER	Not Null	FK to PF_MARKUP_DEFINITION.
LAYOUT_LOCATION	NUMBER	Not Null	The location of this placeholder in the layout. This is used when swapping layouts, as portlets in one layout's location are moved to the other layout's location with the same ID. If the other layout does not have the same number of placeholders, the modulus of the location by number of locations are used.
CREATION_DATE	DATE	Not Null	The date and time the row was created.

Table 8-61 PF_PLACEHOLDER_DEFINITION Table Metadata (Continued)

Column Name	Data Type	Null Value	Description
MODIFIED_DATE	DATE	Not Null	The date and time the row was last modified. This column's data is maintained using a database trigger.
INTERSECTION_ID	NUMBER	Not Null	FK to L10N_INTERSECTION.
LAYOUT_DEFINITION_ID	NUMBER	Not Null	FK to PF_LAYOUT_DEFINITION.

The PF_PLACEMENT Database Table

Each record in this table represents a single placement of a book or portlet on a page.

Table 8-62 PF_PLACEMENT Table Metadata

Column Name	Data Type	Null Value	Description
PLACEMENT_ID	NUMBER	Not Null	PK—A unique, system-generated number to use as the record ID.
PAGE_INSTANCE_ID	NUMBER	Not Null	FK to PF_PAGE_INSTANCE.
POSITION	NUMBER	Not Null	The position within the placeholder where this placement lies. Placeholders can contain more than one placement.
CREATION_DATE	DATE	Not Null	The date and time the row was created.
MODIFIED_DATE	DATE	Not Null	The date and time the row was last modified. This column's data is maintained using a database trigger.
PLACEHOLDER_ DEFINITION_ID	NUMBER	Not Null	FK to PF_PLACEHOLDER_DEFINITION.
PORTLET_INSTANCE_ID	NUMBER	Null	FK to PF_PORTLET_INSTANCE.
BOOK_INSTANCE_ID	NUMBER	Null	FK to PF_BOOK_INSTANCE.

Table 8-62 PF_PLACEMENT Table Metadata (Continued)

Column Name	Data Type	Null Value	Description
DESKTOP_INSTANCE_ID	NUMBER	Null	FK to PF_DESKTOP_INSTANCE. If this placement grouping is an administrator- or end-user-customization, the value is non null and points to the administrator's or user's desktop. If this field is null, it represents the library's view.

The PF_PORTAL Database Table

This table identifies a PORTAL application library resource or template that can be associated with a DESKTOP definition.

Table 8-63 PF_PORTAL Table Metadata

Column Name	Data Type	Null Value	Description
PORTAL_PATH	VARCHAR(40)	Not Null	PK—Partial primary key and partial URL to this portal.
WEBAPP_NAME	VARCHAR(80)	Not Null	PK—Name of the J2EE Web application to which the portal resource is scoped.
CREATION_DATE	DATE	Not Null	The date and time the row was created.
MODIFIED_DATE	DATE	Not Null	The date and time the row was last modified. This column's data is maintained using a database trigger.
INTERSECTION_ID	NUMBER	Not Null	FK to L10N_INTERSECTION.
CONTENT_URI	VARCHAR(255)	Null	Defines an optional URI to be forwarded to when only the portal portion of the URL is supplied. You can use this URL (JSP or .portal) to forward to a default desktop or to display a list of desktops available under this portal.

The PF_PORTLET_CATEGORY Database Table

This table associates a PORTLET CATEGORY resource with a PORTLET DEFINITION.

Table 8-64 PF_PORTLET_CATEGORY Table Metadata

Column Name	Data Type	Null Value	Description
PORTLET_DEFINITION_ID	NUMBER	Not Null	PK—A unique, system-generated number to use as the record ID.
PORTLET_CATEGORY_ DEFINITION_ID	NUMBER	Not Null	FK to PF_PORTLET_CATEGORY_ DEFINITION.
CREATION_DATE	DATE	Not Null	The date and time the row was created.
MODIFIED_DATE	DATE	Not Null	The date and time the row was last modified. This column's data is maintained using a database trigger.

The PF_PORTLET_CATEGORY_DEFINITION Database Table

This table identifies a PORTLET CATEGORY and PORTLET CATEGORY hierarchy resource or template for association with a PORTLET resource.

Table 8-65 PF_PORTLET_CATEGORY_DEFINITION Table Metadata

Column Name	Data Type	Null Value	Description
PORTLET_CATEGORY_ DEFINITION_ID	NUMBER	Not Null	PK—A unique, system-generated number to use as the record ID.
CREATION_DATE	DATE	Not Null	The date and time the row was created.
MODIFIED_DATE	DATE	Not Null	The date and time the row was last modified. This column's data is maintained using a database trigger.
INTERSECTION_ID	NUMBER	Not Null	FK to L10N_INTERSECTION.
WEBAPP_NAME	VARCHAR(80)	Not Null	Name of the J2EE Web application to which the portal resource is scoped.
PARENT_CATEGORY_ DEFINITION_ID	NUMBER	Null	FK to PF_PORTLET_CATEGORY_ DEFINITION that identifies the parent portlet category. NULL if this is a top level category.

The PF_PORTLET_DEFINITION Database Table

This table identifies the characteristics of a PORTLET library resource or template that can used as the user interfaces for a web application.

Table 8-66 PF_PORTLET_DEFINITION Table Metadata

Column Name	Data Type	Null Value	Description
PORTLET_DEFINITION_ID	NUMBER	Not Null	PK—A unique, system-generated number to use as the record ID.
MARKUP_DEFINITION_ID	NUMBER	Not Null	FK to PF_MARKUP_DEFINITION.
IS_PUBLIC	NUMBER	Not Null	A boolean indicating that the portlet definition is public. Only public portlet definitions are ever exposed to "visitors."
IS_FORKABLE	NUMBER	Not Null	A boolean indicating that the portlet supports multi-threading.
FORK_RENDER	NUMBER	Not Null	A boolean indicating whether multi-threading is being used for this portlet; this value can be true only if IS_FORKABLE is true.
IS_CACHEABLE	NUMBER	Not Null	A boolean indicating whether this portlet can use render caching.
CACHE_EXPIRES	NUMBER	Not Null	Indicates whether this portlet is using caching and if so, gives the ttl: -1 indicates off; 0..n indicates a ttl for the cache. Can have a value other than -1 only if IS_CACHEABLE is true.

Table 8-66 PF_PORTLET_DEFINITION Table Metadata (Continued)

Column Name	Data Type	Null Value	Description
IS_PORTLET_FILE_DELETED	NUMBER	Not Null	A boolean that indicates whether the PORTLET_FILE associated with this object has been removed from the file system. This flag is set to true only when the .portlet file is deleted and the portlet is still in use. You can either return the .portlet file and this flag is automatically reset, or remove the portlet in the WebLogic Administration Portal.
CREATION_DATE	DATE	Not Null	The date and time the row was created.
MODIFIED_DATE	DATE	Not Null	The date and time the row was last modified. This column's data is maintained using a database trigger.
PORTLET_LABEL	VARCHAR(80)	Not Null	A moniker used to reference this portal resource for development purposes.
WEBAPP_NAME	VARCHAR(80)	Not Null	Name of the J2EE Web application to which the portal resource is scoped.
CONTENT_URI	VARCHAR(255)	Not Null	The content URI for this portlet (JSP, HTML). This value can be null for Java (JSR168) portlets.
EDIT_URI	VARCHAR(255)	Null	The Edit mode URI (JSP) for this portlet (if the portlet supports edit mode).
HELP_URI	VARCHAR(255)	Null	The Help mode URI (JSP) for this portlet (if the portlet supports help mode).
BACKING_FILE	VARCHAR(255)	Null	The optional backing file (Java class name) for this portlet. Backing classes must implement JspBacking or extend AbstractJspBacking.

Table 8-66 PF_PORTLET_DEFINITION Table Metadata (Continued)

Column Name	Data Type	Null Value	Description
PORTLET_FILE	VARCHAR(255)	Null	The (*.portlet) file describing the controls that make up the portlet.

The PF_PORTLET_INSTANCE Database Table

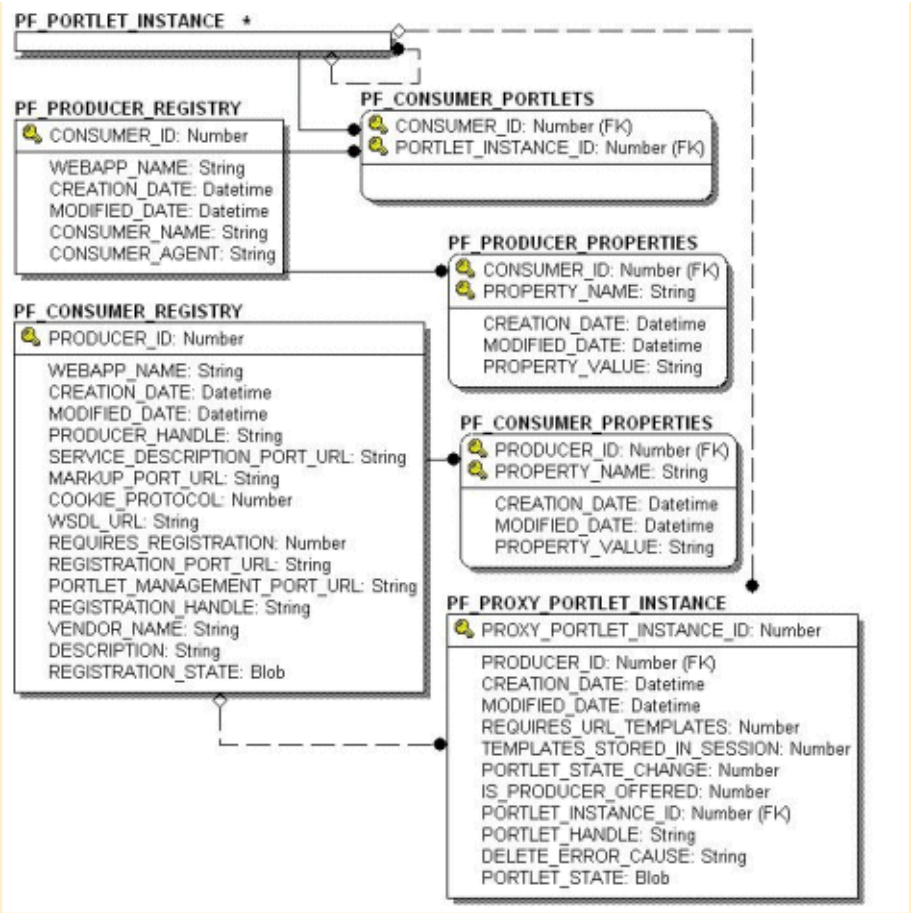
This table identifies a customized or localized instance of a portlet.

Table 8-67 PF_PORTLET_INSTANCE Table Metadata

Column Name	Data Type	Null Value	Description
PORTLET_INSTANCE_ID	NUMBER	Not Null	PK—A unique, system-generated number to use as the record ID.
PORTLET_DEFINITION_ID	NUMBER	Not Null	FK to PF_PORTLET_DEFINITION.
DEFAULT_MINIMIZED	NUMBER	Not Null	A boolean that indicates whether the portlet is to be displayed in the minimized state by default.
INSTANCE_TYPE	NUMBER	Not Null	Type codes for the portlet instance. Valid values: 1=Primary, 3=Admin, 4=User.
CREATION_DATE	DATE	Not Null	The date and time the row was created.
MODIFIED_DATE	DATE	Not Null	The date and time the row was last modified. This column's data is maintained using a database trigger.
TITLE_BAR_ORIENTATION	NUMBER	Null	A hint to the skeleton file to display this portlet's title bar in either the top, left, right, or bottom location. Not all skeletons may implement this and therefore may not have any effect.
INTERSECTION_ID	NUMBER	Not Null	FK to L10N_INTERSECTION.
PORTLET_LABEL	VARCHAR(80)	Not Null	A moniker used to reference this portal resource for development purposes.
THEME_DEFINITION_ID	NUMBER	Null	FK to PF_THEME_DEFINITION.
PARENT_PORTLET_ INSTANCE_ID	NUMBER	Null	FK to PF_PORTLET_INSTANCE that identifies the parent portlet instance. the value is null if this is a top-level portlet instance.

WSRP (Web Services for Remote Portlets) Objects

Figure 8-11 Entity-Relation Diagram for the WSRP Tables



The PF_CONSUMER_PORTLETS Database Table

The PF_CONSUMER_PORTLETS table associates consumer IDs and portlet instance IDs so that when a consumer de-registers from a producer, the producer can clean up any portlets that it created for the consumer.

Table 8-68 The PF_CONSUMER_PORTLETS Database Table

Column Name	Data Type	Null Value	Description
CONSUMER_ID	NUMBER	Not Null	PK/FK to PF_PRODUCER_REGISTRY.
PORTLET_ID	NUMBER	Not Null	PK/FK to PF_PORTLET_INSTANCE.

The PF_CONSUMER_PROPERTIES Database Table

This table contains optional registration properties. You can set up the consumer to ask for these during registration.

Table 8-69 The PF_CONSUMER_PROPERTIES Database Table

Column Name	Data Type	Null Value	Description
PRODUCER_ID	INTEGER	Not Null	PK/FK to PF_PRODUCER_REGISTRY.
PROPERTY_NAME	VARCHAR (80)	Not Null	PK—The name of the property.
CREATION_DATE	DATE	Not Null	The date and time the row was created.
MODIFIED_DATE	DATE	Not Null	The date and time the row was modified. The column's data is maintained using a database trigger.
PROPERTY_VALUE	VARCHAR (80)	Null	The value associated with the property name.

The PF_CONSUMER_REGISTRY Database Table

This table contains registration handles that are assigned by the producer during registration by a consumer.

Table 8-70 The PF_CONSUMER_REGISTRY Database Table

Column Name	Data Type	Null Value	Description
PRODUCER_ID	INTEGER	Not Null	PK—A unique, system-generated number to use as the record ID.
WEBAPP_NAME	VARCHAR (80)	Not Null	Name of the J2EE Web application (as defined in `config.xml`) to which the portal application is scoped.
CREATION_DATE	DATE	Not Null	The date and time the row was created.
MODIFIED_DATE	DATE	Not Null	The date and time the row was modified. The column's data is maintained using a database trigger.
PRODUCER_HANDLE	VARCHAR (40)	Not Null	Uniquely identifies the producer to the consumer.
SERVICE_DESCRIPTION_PORT_URL	VARCHAR (255)	Not Null	URL to the description service port offered by the producer.
MARKUP_PORT_URL	VARCHAR (255)	Not Null	URL to the markup service port offered by the producer.
COOKIE_PROTOCOL	SMALLINT	Not Null	The cookie protocol. Values: 0 = None, 1 = Per User, 2 = Per Group.
WSDL_URL	VARCHAR (255)	Not Null	URL to the WSDL offered by the producer.
REQUIRES_REGISTRATION	SMALLINT	Not Null	A boolean indicating that registration is required.
REGISTRATION_PORT_URL	VARCHAR (255)	Null	URL to the registration service port offered by the producer (if offered).
PORTLET_MANAGEMENT_PORT_URL	VARCHAR (255)	Null	URL to the portlet management service port offered by the producer (if offered).
REGISTRATION_HANDLE	VARCHAR (255)	Null	Registration handle returned by the producer after registration.

Table 8-70 The PF_CONSUMER_REGISTRY Database Table

Column Name	Data Type	Null Value	Description
VENDOR_NAME	VARCHAR (255)	Null	Name of the vendor of the producer implementation.
DESCRIPTION	VARCHAR (255)	Null	A description of the portlet.
REGISTRATION_STATE	BLOB	Null	Registration state returned by the producer after registration.

The PF_PRODUCER_PROPERTIES Database Table

This table contains optional registration properties. The producer might be asked for these during registration.

Table 8-71 The PF_PRODUCER_PROPERTIES Database Table

Column Name	Data Type	Null Value	Description
CONSUMER_ID	INTEGER	Not Null	PK/FK to PF_CONSUMER_REGISTRY.
PROPERTY_NAME	VARCHAR (80)	Not Null	PK—The name of the property.
CREATION_DATE	DATE	Not Null	The date and time the row was created.
MODIFIED_DATE	DATE	Not Null	The date and time the row was modified. This column's data is maintained using a database trigger.
PROPERTY_VALUE	VARCHAR (80)	Null	The value associated with the PROPERTY_NAME.

The PF_PRODUCER_REGISTRY Database Table

This table contains producer-generated registration handles stored for each consumer during registration.

Table 8-72 The PF_PRODUCER_REGISTRY Database Table

Column Name	Data Type	Null Value	Description
CONSUMER_ID	INTEGER	Not Null	A unique system-generated number to use as the record ID.
WEBAPP_NAME	VARCHAR (80)	Not Null	Name of the J2EE Web application to which the portal application is scoped.
CREATION_DATE	DATE	Not Null	The date and time the row was created.
MODIFIED_DATE	DATE	Not Null	The date and time the row was modified. This column's data is maintained using a database trigger.
CONSUMER_NAME	VARCHAR (80)	Null	A unique name that identifies the consumer. For the producer to assert user identity, the consumer name must correspond to the alias of the consumer's public key deployed in the producer's key store.
CONSUMER_AGENT	VARCHAR (80)	Null	Name and version of the consumer's vendor. The value must start with `productName.majorVersion.minorVersion` where `productName` identifies the product that the consumer installed for its deployment, and `majorVersion` and `minorVersion` are vendor-defined indications of the version of its product.

The PF_PROXY_PORTLET_INSTANCE Database Table

The consumer manages remote portlet-specific data from here. The framework inserts data into this table whenever a proxy portlet instance is created (including successors). When portlet instances are deleted, the IS_SET_FOR_DESTROY flag is set for subsequent cleanup.

Table 8-73 The PF_PROXY_PORTLET_INSTANCE Database Table

Column Name	Data Type	Null Value	Description
PROXY_PORTLET_INSTANCE_ID	INTEGER	Not Null	A unique system-generated number to use as the record ID.
PRODUCER_ID	INTEGER	Not Null	FK to PF_CONSUMER_REGISTRY.
CREATED_DATE	DATE	Not Null	The date and time the row was created.
MODIFIED_DATE	DATE	Not Null	The date and time the row was modified. This column's data is maintained using a database trigger.
REQUIRES_URL_TEMPLATES	SMALLINT	Not Null	A boolean indicating that URL templates are required by the producer.
TEMPLATES_STORED_IN_SESSION	SMALLINT	Not Null	A boolean indicating whether the consumer should send templates with every request. The default is 1 = True.
PORTLET_STATE_CHANGE	SMALLINT	Not Null	A flag that indicates how the consumer handles customizations of remote portlets. Based on the value of this flag, the consumer may clone remote portlets. Valid values include: 0 = Readonly (default) 1 = CBW(CloneBeforeWrite) 2 = Read/Write.

Table 8-73 The PF_PROXY_PORTLET_INSTANCE Database Table

Column Name	Data Type	Null Value	Description
IS_PRODUCER_OFFERED	NUMBER	Not Null	Identifies the portlet as producer-offered. The defaults is 1 = True. If IS_PRODUCER_OFFERED is True, and the PORTLET_INSTANCE_ID is null, this PF_PROXY_PORTLET_INSTANCE is removed from the database during data cleanup processing.
PORTLET_INSTANCE_ID	INTEGER	Null	FK to PF_PORTLET_INSTANCE for the proxy portlet. If the associated PF_PORTLET_INSTANCE row is deleted, the value of this column is set to null.
PORTLET_HANDLE	VARCHAR (255)	Null	The handle to the remote portlet as it is specified by the producer. The consumer uses portlet handles throughout the communication to address and interact with portlets using the producer.
DELETE_ERROR_CAUSE	VARCHAR (255)	Null	A description of the cause of the error, if an error is encountered while trying to delete the counter part of this proxy portlet on the producer.
PORTLET_STATE	BLOB	Null	Portlet state as returned by the producer after implicit/explicit cloning.

The PF_PORTLET_PREFERENCE Database Table

This table identifies preference values for the portlet instance.

Table 8-74 PF_PORTLET_PREFERENCE Table Metadata

Column Name	Data Type	Null Value	Description
PORTLET_INSTANCE_ID	NUMBER	Not Null	PK—A unique, system-generated number to use as the record ID.

Table 8-74 PF_PORTLET_PREFERENCE Table Metadata (Continued)

Column Name	Data Type	Null Value	Description
PREFERENCE_NAME	VARCHAR(40)	Not Null	An optional name associated with the preference values.
CREATION_DATE	DATE	Not Null	The date and time the row was created.
MODIFIED_DATE	DATE	Not Null	The date and time the row was last modified. This column's data is maintained using a database trigger.
IS_MODIFIABLE	NUMBER	Not Null	A boolean, indicating whether the name/value of this preference can be modified by portlets.
IS_MULTIVALUED	NUMBER	Not Null	A boolean, indicating whether a preference can have more than one value.
PREFERENCE_DESCRIPTION	VARCHAR(255)	Null	An optional description of the portlet preferences.

The PF_PORTLET_PREFERENCE_VALUE Database Table

This table maintains values of portlet preferences. There is a one-to-many correspondence between the records in the PF_PORTLET_PREFERENCE table and this table.

Table 8-75 PF_PORTLET_PREFERENCE_VALUE Table Metadata

Column Name	Data Type	Null Value	Description
PORTLET_PREFERENCE_VALUE_ID	NUMBER	Not Null	PK—A unique, system-generated number to use as the record ID.
PORTLET_INSTANCE_ID	NUMBER	Not Null	FK to PF_PORTLET_PREFERENCE.
PREFERENCE_NAME	VARCHAR(40)	Not Null	FK to PF_PORTLET_PREFERENCE.
CREATION_DATE	DATE	Not Null	The date and time the row was created.
MODIFIED_DATE	DATE	Not Null	The date and time the row was last modified. This column's data is maintained using a database trigger.
PREFERENCE_VALUE	VARCHAR(255)	Null	The actual value for this preference.

The PF_SHELL_DEFINITION Database Table

This table represents a shell definition. There is a one-to-one correspondence between records in this table and .shell files.

Table 8-76 PF_SHELL_DEFINITION Table Metadata

Column Name	Data Type	Null Value	Description
SHELL_DEFINITION_ID	NUMBER	Not Null	PK—A unique, system-generated number to use as the record ID.
MARKUP_DEFINITION_ID	NUMBER	Not Null	FK to PF_MARKUP_DEFINITION.
CREATION_DATE	DATE	Not Null	The date and time the row was created.
MODIFIED_DATE	DATE	Not Null	The date and time the row was last modified. This column's data is maintained using a database trigger.
INTERSECTION_ID	NUMBER	Not Null	FK to L10N_INTERSECTION.
WEBAPP_NAME	VARCHAR(80)	Not Null	Name of the J2EE Web application to which the portal resource is scoped.
IS_SHELL_FILE_DELETED	NUMBER	Not Null	A boolean indicating that the file associated with this shell was removed from the file system. If the shell is not being used, then the record is deleted outright. This flag is set to true only when the .shell file is deleted and the shell is still in use. You can either return the .shell file and this flag is automatically reset, or remove the shell with a replacement in the WebLogic Administration Portal.
SHELL_FILE	VARCHAR(255)	Not Null	The name of the .shell file contained in the application's framework/markup/shell directory backing this shell definition.

The PF_THEME_DEFINITION Database Table

This table represents a theme definition. There is a one-to-one correspondence between records in this table and `.theme` files.

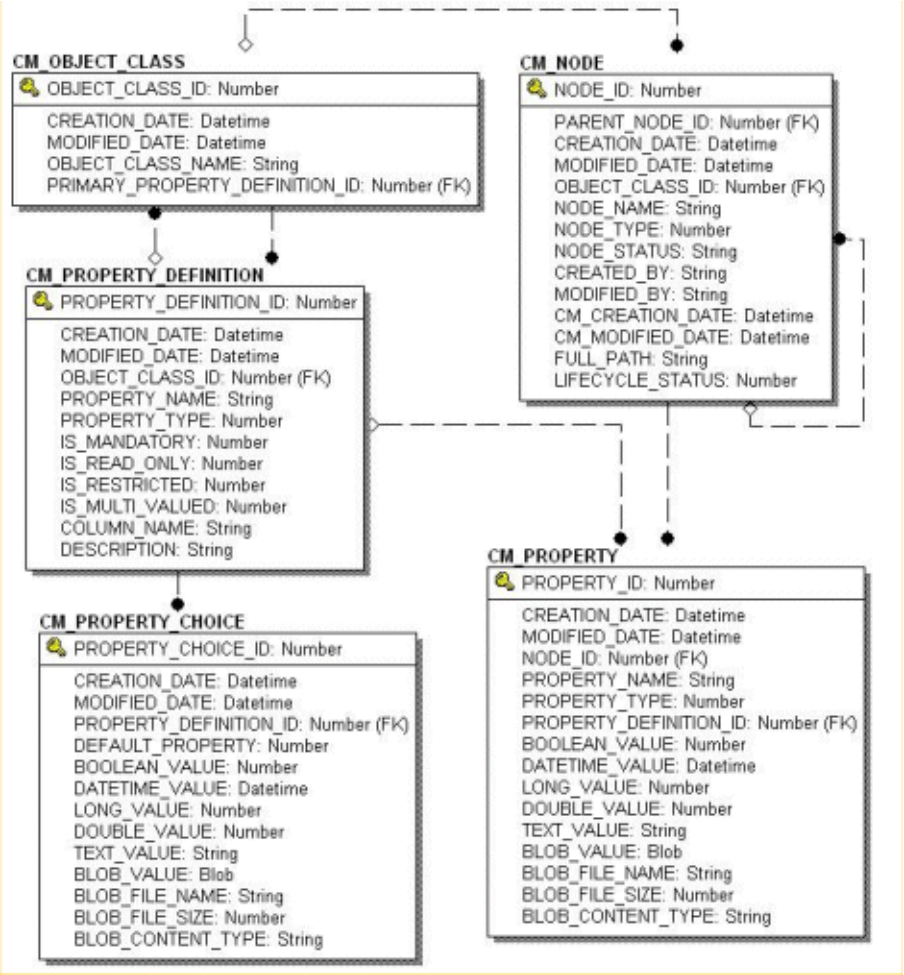
Table 8-77 PF_THEME_DEFINITION Table Metadata

Column Name	Data Type	Null Value	Description
THEME_DEFINITION_ID	NUMBER	Not Null	PK—A unique, system-generated number to use as the record ID.
CREATION_DATE	DATE	Not Null	The date and time the row was created.
MODIFIED_DATE	DATE	Not Null	The date and time the row was last modified. This column's data is maintained using a database trigger.
INTERSECTION_ID	NUMBER	Not Null	FK to L10N_INTERSECTION.
MARKUP_DEFINITION_ID	NUMBER	Not Null	FK to PF_MARKUP_DEFINITION.
WEBAPP_NAME	VARCHAR(80)	Not Null	Name of the J2EE Web application to which the portal resource is scoped.
IS_THEME_FILE_DELETED	NUMBER	Not Null	A boolean indicating that the file associated with this theme was removed from the file system. If the theme is not being used, then the record is deleted outright. This flag is set to true only when the `.theme` file is deleted and the theme is still in use. You can either return the `.theme` file and this flag is automatically reset, or remove the theme using the WebLogic Administration Portal.
THEME_FILE	VARCHAR(255)	Not Null	The name of the `.theme` file contained in the application's framework/markup/theme directory backing this theme definition.

Content Management Database Objects

Figure 8-12 shows the logical entity-relation diagram for the WebLogic Portal Content Management tables.

Figure 8-12 Entity-Relation Diagram for the Content Management Tables



The Content Management Data Dictionary Tables

The Content Management system has the following tables:

- The CM_NODE Database Table
- The CM_OBJECT_CLASS Database Table
- The CM_PROPERTY Database Table
- The CM_PROPERTY_CHOICE Database Table
- The CM_PROPERTY_DEFINITION Database Table

The CM_NODE Database Table

In the `CM_NODE` table, a node represents an element in a hierarchy that can either be a "Hierarchy Node" or a "Content Node." A hierarchy node can contain both other hierarchy and content nodes while a content node can contain only other content nodes. Nodes can contain properties based on the `ObjectClass` (schema) defined for it.

Both Content Nodes and Hierarchy Nodes can contain an ObjectClass and properties. Each node has a path that uniquely identifies it within the repository.

Table 8-78 CM_NODE Table Metadata

Column Name	Data Type	Null Value	Description
NODE_ID	NUMBER	Not Null	PK—A unique, system-generated number to use as the record ID.
PARENT_NODE_ID	NUMBER	Null	FK—The node's parent record ID (NODE_ID).
CREATION_DATE	DATE	Not Null	The date and time the row was created.
MODIFIED_DATE	DATE	Not Null	The date and time the row was last modified. This column's data is maintained using a database trigger.
OBJECT_CLASS_ID	NUMBER	Null	FK—The object class ID associated to the node.

Table 8-78 CM_NODE Table Metadata (Continued)

Column Name	Data Type	Null Value	Description
NODE_NAME	VARCHAR(50)	Not Null	The name of the node. The name is unique relative to its siblings. The name must not contain forward or backward slashes.
NODE_TYPE	NUMBER	Not Null	The node type. Either 1 for Hierarchy Node or 2 for Content Node.
NODE_STATUS	VARCHAR(40)	Null	The status of the node. The available values are defined by the application as property definition choices.
CREATED_BY	VARCHAR(100)	Not Null	The ID of the user that created the node.
MODIFIED_BY	VARCHAR(100)	Null	The ID of the user that last modified the node.
CM_CREATION_DATE	DATE	Not Null	The date and time the row was created. Maintained by the application.
CM_MODIFIED_DATE	DATE	Not Null	The date and time the row was last modified. Maintained by the application.
FULL_PATH	VARCHAR(254)	Null	AK—Each node has a path that uniquely identifies it within the repository. The path is defined in a UNIX-like format such as /a/b/c where "/" is the root and "a" ("a" is the Nodes NODE_NAME) is the root's child. The path must always begin with "/" and never end with it. So neither of the following are valid: a/b/c/d or /a/b/d/d/.
LIFECYCLE_STATUS	INTEGER	Null	The specific lifecycle status that the node version has been assigned:

The CM_OBJECT_CLASS Database Table

The ObjectClass is the schema for a Node. It has both an ID and a name that uniquely identifies it within a content repository. An ObjectClass can have PropertyDefinitions associated with it

that define the shape of Properties required for a Node. This does not mean that the Property must contain a value, but simply that the Property must exist for the Node.

The ObjectClass may have a primary PropertyDefinition that defines the primary content Property for a Node. This allows for the definition of content in the schema since the schema does not distinguish between content and meta-content. A Node is considered valid in the repository only if its Properties conform to its ObjectClass PropertyDefinitions.

Table 8-79 CM_OBJECT_CLASS Table Metadata

Column Name	Data Type	Null Value	Description
OBJECT_CLASS_ID	NUMBER	Not Null	PK—A unique, system-generated number to use as the record ID.
CREATION_DATE	DATE	Not Null	The date and time the row was created.
MODIFIED_DATE	DATE	Not Null	The date and time the row was last modified. This column's data is maintained using a database trigger.
OBJECT_CLASS_NAME	VARCHAR(100)	Not Null	AK—A unique name for the object class.
PRIMARY_PROPERTY_ DEFINITION_ID	NUMBER	Null	FK—The PROPERTY_DEFINITION_ID for the primary CM_PROPERTY_ DEFINITION table row that defines the content for a node associated to the object class.

The CM_PROPERTY Database Table

The CM_PROPERTY table identifies a property. The property consists of a name value pair; the name is unique relative to the CM_NODE, and the value is either a Date, BLOB, Boolean, Number, Float, or Varchar.

Only one value should be set on a given row; if the value is a BLOB, then all of the BLOB_ columns can be set. If the IS_MULTIVALUED column is set to 1, then there will be multiple rows with the same property name and same NODE_ID. A property can represent both the content and meta-content for a Node.

Table 8-80 CM_PROPERTY Table Metadata

Column Name	Data Type	Null Value	Description
PROPERTY_ID	NUMBER	Not Null	PK—A unique, system-generated number to use as the record ID.
CREATION_DATE	DATE	Not Null	The date and time the row was created.
MODIFIED_DATE	DATE	Not Null	The date and time the row was last modified. This column's data is maintained using a database trigger.
NODE_ID	NUMBER	Not Null	FK—The ID of the node that contains the property.
PROPERTY_NAME	VARCHAR(100)	Not Null	The name of the property. It must be unique relative to its node.
PROPERTY_TYPE	NUMBER	Not Null	The type of the property: BOOLEAN = 0; NUMBER = 1; FLOAT = 2; VARCHAR = 3; DATE = 4; BLOB = 5.
PROPERTY_DEFINITION_ID	NUMBER	Null	FK—The ID of the property definition to which this property must conform.
BOOLEAN_VALUE	NUMBER	Null	True (1) for the Property if the PROPERTY_TYPE is Boolean (PROPERTY_TYPE=0).
DATETIME_VALUE	DATE	Null	The datetime value for the Property if the PROPERTY_TYPE is DATE (PROPERTY_TYPE=4).
LONG_VALUE	NUMBER	Null	The long number or integer value for the Property if the PROPERTY_TYPE is NUMBER(PROPERTY_TYPE=1).
DOUBLE_VALUE	FLOAT	Null	The floating point decimal number value for the Property if the PROPERTY_TYPE is FLOAT(PROPERTY_TYPE=2).
TEXT_VALUE	VARCHAR(254)	Null	The textual property value for the Property if the PROPERTY_TYPE is VARCHAR(PROPERTY_TYPE=3).

Table 8-80 CM_PROPERTY Table Metadata (Continued)

Column Name	Data Type	Null Value	Description
BLOB_VALUE	BLOB	Null	The binary large object for the Property if the PROPERTY_TYPE is BLOB(PROPERTY_TYPE=5).
BLOB_FILE_NAME	VARCHAR(50)	Null	The name of the file associated with the BLOB_VALUE.
BLOB_FILE_SIZE	NUMBER	Null	The size of the file in bytes associated with the BLOB_VALUE.
BLOB_CONTENT_TYPE	VARCHAR(100)	Null	The content type (mime type and characterset) for the BLOB_VALUE. For example: “text/html;charset=iso8859-1”

The CM_PROPERTY_CHOICE Database Table

This table identifies the valid values or choices for a PropertyDefinition (row in the CM_PROPERTY_DEFINITION table). A property choice can identify a default choice (DEFAULT_PROERTY=1); if the creator of a Property does not choose different values, it is set as a Property value.

If the PropertyChoice value is defined as NULL (no value is supplied for the PROPERTY_TYPE), it allows for an empty choice. For example, a Property that has a String type (or TEXT_VALUE) could have three PropertyChoices - “blue,” “red,” “*,” and null.

Table 8-81 CM_PROPERTY_CHOICE Table Metadata

Column Name	Data Type	Null Value	Description
PROPERTY_CHOICE_ID	NUMBER	Not Null	PK—A unique, system-generated number to use as the record ID.
CREATION_DATE	DATE	Not Null	The date and time the row was created.
MODIFIED_DATE	DATE	Not Null	The date and time the row was last modified. This column’s data is maintained using a database trigger.
PROPERTY_DEFINITION_ID	NUMBER	Not Null	FK—The ID of the property definition that contains the property choice.

Table 8-81 CM_PROPERTY_CHOICE Table Metadata (Continued)

Column Name	Data Type	Null Value	Description
DEFAULT_PROPERTY	NUMBER	Not Null	Set to 1 if the property choice is a default, or 0 if it is not.
BOOLEAN_VALUE	NUMBER	Null	True (1) for the Property if the PROPERTY_TYPE is BOOLEAN (PROPERTY_TYPE=0).
DATETIME_VALUE	DATE	Null	The date/time value for the Property if the PROPERTY_TYPE is DATE (PROPERTY_TYPE=4).
LONG_VALUE	NUMBER	Null	The long number or integer value for the Property if the PROPERTY_TYPE is NUMBER(PROPERTY_TYPE=1).
DOUBLE_VALUE	FLOAT	Null	The floating point decimal number value for the Property if the PROPERTY_TYPE is FLOAT(PROPERTY_TYPE=2).
TEXT_VALUE	VARCHAR(254)	Null	The textual property value for the Property if the PROPERTY_TYPE is VARCHAR(PROPERTY_TYPE=3).
BLOB_VALUE	BLOB	Null	The binary large object for the Property if the PROPERTY_TYPE is BLOB(PROPERTY_TYPE=5).
BLOB_FILE_NAME	VARCHAR(50)	Null	The name of the file associated with the BLOB_VALUE.
BLOB_FILE_SIZE	NUMBER	Null	The size of the file in bytes associated with the BLOB_VALUE.
BLOB_CONTENT_TYPE	VARCHAR(100)	Null	The content type (mime type and characterset) for the BLOB_VALUE. For example: “text/html;charset=iso8859-1”

The CM_PROPERTY_DEFINITION Database Table

The PropertyDefinition table defines the shape of a property. It describes the property type (BLOB, Boolean, Varchar, Float, Date, Number), whether it is required, whether it is editable,

the default value, and restricted values, if applicable. A PropertyDefinition can have 0..n PropertyChoices.

This is a list of values that you can select for a Property's values. Rules for a PropertyDefinition are as follows:

- If the PropertyDefinition contains a reference, it cannot be multi-valued or binary.
- If the PropertyDefinition is binary, it cannot be multi-valued or restricted and can have only one PropertyChoice.
- If the PropertyDefinition is boolean, it cannot be multi-valued. If the PropertyDefinition is restricted, then the Property's value(s) must be contained in the PropertyChoice list, or be null.

For example: consider a PropertyDefinition named “color”. It has PropertyChoices “blue,” “green,” and “red”. If the PropertyDefinition is restricted then the value of a Property defined by this PropertyDefinition cannot have a value that isn't “green,” “red,” “blue,” or null.

Table 8-82 CM_PROPERTY_DEFINITION Table Metadata

Column Name	Data Type	Null Value	Description
PROPERTY_DEFINITION_ID	NUMBER	Not Null	PK—A unique, system-generated number to use as the record ID.
CREATION_DATE	DATE	Not Null	The date and time the row was created.
MODIFIED_DATE	DATE	Not Null	The date and time the row was last modified. This column’s data is maintained using a database trigger.
OBJECT_CLASS_ID	NUMBER	Not Null	FK-The OBJECT_CLASS_ID of the property definitions CM_OBJECT_CLASS.
PROPERTY_NAME	VARCHAR(100)	Not Null	The name associated with the property definition. The combination of PROPERTY_NAME and OBJECT_ CLASS_ID for an Alternate Key for the CM_PROPERTY_DEFINITION table.
PROPERTY_TYPE	NUMBER	Not Null	The type of the property: BOOLEAN = 0; NUMBER = 1; FLOAT = 2; VARCHAR = 3; DATE = 4; BLOB = 5.

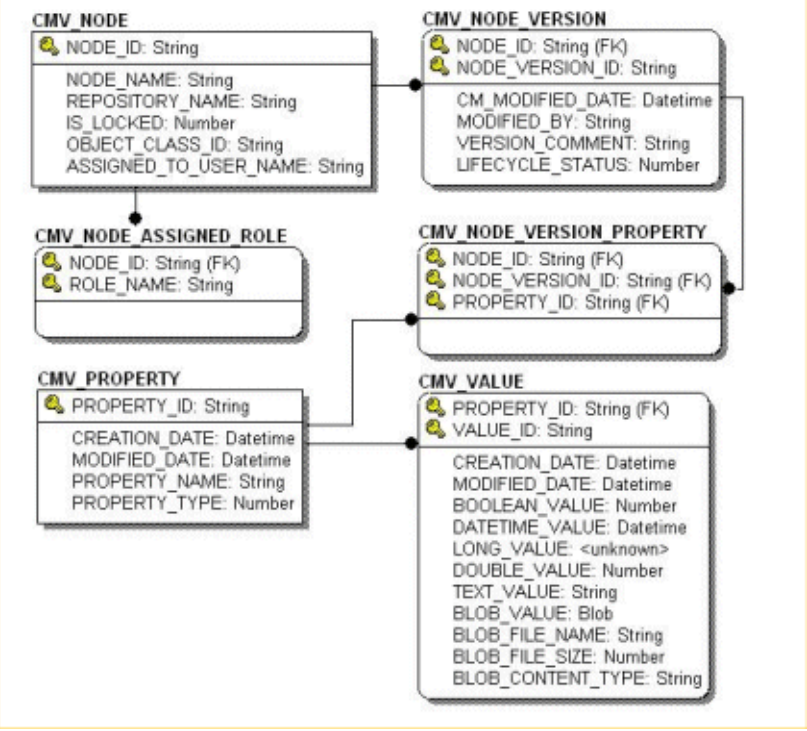
Table 8-82 CM_PROPERTY_DEFINITION Table Metadata (Continued)

Column Name	Data Type	Null Value	Description
IS_MANDATORY	NUMBER	Not Null	True if the value of a property must be set.
IS_READ_ONLY	NUMBER	Not Null	True if the value of a property should not be set by an end user.
IS_RESTRICTED	NUMBER	Not Null	True if the value of a property should come from the property choice values.
IS_MULTI_VALUED	NUMBER	Not Null	True if there can be multiple rows with the same property name, node_id, but different property IDs.
COLUMN_NAME	VARCHAR(30)	Null	The name of a column added to the CM_NODE table that defines an explicit property.
DESCRIPTION	VARCHAR(254)	Null	A description of the property definition.

Content Management Virtual Database Objects

Figure 8-13 shows the logical entity-relation diagram for the WebLogic Portal Content Management tables.

Figure 8-13 Entity-Relation Diagram for the Content Management Virtual Tables



The CMV_NODE Table

This table uniquely identifies a content-managed node from a BEA repository (that is, CM_NODE table) that has been versioned and is being edited within the Content Management Virtual Repository.

Table 8-83 The CMV_NODE Table

Column Name	Data Type	Null Value	Description
NODE_ID	VARCHAR (254)	Not Null	PK—A unique, system-generated number to use as the record ID.
REPOSITORY_NAME	VARCHAR (254)	Not Null	The name of the repository where the node was created and published.
IS_LOCKED	SMALLINT	Not Null	Flag to determine if the record is locked.

Table 8-83 The CMV_NODE Table

Column Name	Data Type	Null Value	Description
OBJECT_CLASS_ID	VARCHAR (254)	Not Null	The object class ID that is associated with the node.
OBJECT_CLASS_ID	NUMBER	Null	FK—The object class ID associated to the node.
ASSIGNED_TO_USER_NAME	VARCHAR(200)	Null	Username to which the node is assigned.

The CMV_NODE_ASSIGNED_ROLE Table

This table uniquely identifies all roles for a given node that have authorization to view or alter the node.

Table 8-84 The CMV_NODE_ASSIGNED_ROLE Table

Column Name	Data Type	Null Value	Description
NODE_ID	VARCHAR (254)	Not Null	PK/FK—the ID of the node that roles are associated with. Foreign key relationship to CMV_NODE table.
ROLE_NAME	VARCHAR (254)	Not Null	PK—the name of the role.

The CMV_NODE_VERSION Table

This table uniquely identifies all the versions of a mode within the Content Management Virtual Repository.

Table 8-85 The CMV_NODE_VERSION Table

Column Name	Data Type	Null Value	Description
NODE_ID	VARCHAR (254)	Not Null	PK/FK—the ID of the node for which versions have been created. Foreign key relationship to CMV_NODE table.
NODE_VERSION_ID	VARCHAR (254)	Not Null	PK—the unique version ID for the node.
CM_MODIFIED_DATE	DATE	Not Null	Date the node version was last edited.
MODIFIED_BY	VARCHAR (254)	Not Null	Username of the person who last edited the node version.
VERSION_COMMENT	VARCHAR (254)	Not Null	Comment added to a node version when saving.
LIFECYCLE_STATUS	INTEGER	Null	Specific lifecycle status that the node version has been assigned (for example, In Progress, Published, and so on).

The CMV_PROPERTY Table

This table uniquely identifies a property that can be associated with a node version. For example, some properties of a book might be author, title, and subject.

Table 8-86 The CMV_PROPERTY Table

Column Name	Data Type	Null Value	Description
PROPERTY_ID	VARCHAR (254)	Not Null	PK—A unique, system-generated number to use as the record ID.
CREATION_DATE	DATE	Not Null	The date and time the row was created.

Table 8-86 The CMV_PROPERTY Table

Column Name	Data Type	Null Value	Description
MODIFIED_DATE	DATE	Not Null	Date and time the row was last modified. This column's data is maintained using a database trigger.
PROPERTY_NAME	VARCHAR (100)	Not Null	The name of the property. It must be unique relative to its node.
PROPERTY_TYPE	SMALLINT	Not Null	The type of the property: BOOLEAN = 0 NUMBER = 1 FLOAT = 2 VARCHAR = 3 DATE = 4 BLOB = 5.

The CMV_VALUE Table

This table uniquely identifies a value for a given property. For example, a property SUBJECT for a BOOK might have a value of FINANCE.

Only one value is set on a given record. If the value is BLOB, then all the BLOB_ columns can be set.

Table 8-87 The CMV_VALUE Table

Column Name	Data Type	Null Value	Description
PROPERTY_ID	VARCHAR (254)	Not Null	PK/FK—ID of the property with which the values are associated. Foreign key relationship to CMV_PROPERTY.
VALUE_ID	VARCHAR (254)	Not Null	PK—A unique, system-generated number to use as the value ID.

Table 8-87 The CMV_VALUE Table

Column Name	Data Type	Null Value	Description
CREATION_DATE	DATE	Not Null	The date and time the row was created
			Date and time the row was last modified. This column's data is maintained using a database trigger.
MODIFIED_DATE	DATE	Not Null	Date and time the row was last modified. This column's data is maintained using a database trigger.
BOOLEAN_VALUE	SMALLINT	Not Null	Flag to determine if property is a Boolean value: 1= True, 0 = False.
DATETIME_VALUE	DATE	Null	The datetime value for the property if the PROPERTY_TYPE is DATE.
LONG_VALUE	NUMERIC (20)	Null	The long number or integer value for the property if the PROPERTY_TYPE is NUMBER.
DOUBLE_VALUE	FLOAT	Null	The floating point decimal number value for the property value if the PROPERTY_TYPE is FLOAT.
TEXT_VALUE	VARCHAR (254)	Null	The textual property value if the PROPERTY_TYPE is VARCHAR.
BLOB_VALUE	BLOB	Null	The binary large object for the property value if the PROPERTY_TYPE is BLOB.
BLOB_FILE_NAME	VARCHAR (50)	Null	The name of the file associated with BLOB_VALUE.
BLOB_FILE_SIZE	INTEGER	Null	The size of the file (in bytes) associated with BLOB_VALUE.
BLOB_CONTENT_TYPE	VARCHAR (100)	Null	The content type (MIME and character set) for the BLOB_VALUE. For example: "text/html;charset=iso8859-1"

The CMV_NODE_VERSION_PROPERTY Table

This table uniquely identifies a relationship between a CMV_NODE_VERSION and CMV_PROPERTY.

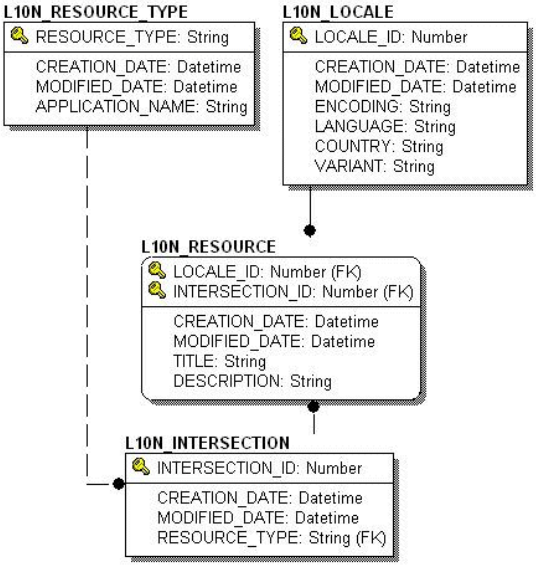
Table 8-88 The CMV_NODE_VERSION_PROPERTY Table

Column Name	Data Type	Null Value	Description
NODE_ID	VARCHAR (254)	Not Null	PK/FK—ID of the node with which the properties are associated. Foreign key relationship to CMV_NODE_VERSION.
NODE_VERSION_ID	VARCHAR (254)	Not Null	PK/FK—ID of the node version with which the properties are associated. Foreign key relationship to CMV_NODE_VERSION.
PROPERTY_ID	VARCHAR (254)	Not Null	PK/FK—ID of the property with which the node versions are associated. Foreign key relationship to CMV_PROPERTY.

Localization Database Objects

This section documents the database objects for the WebLogic Portal package. Figure 8-14 shows the entity-relation diagram for the WebLogic Portal Localization database objects.

Figure 8-14 Entity-Relation Diagram for the Localization Tables



The Localization Dictionary Tables

The following tables support localization:

- The L10N_INTERSECTION Database Table
- The L10N_LOCALE Database Table
- The L10N_RESOURCE Database Table
- The L10N_RESOURCE_TYPE Database Table

The L10N_INTERSECTION Database Table

This table is used to tie an application resource (menu, portlet, and so on) to a localized title and description.

Table 8-89 L10N_INTERSECTION Table Metadata

Column Name	Data Type	Null Value	Description
INTERSECTION_ID	NUMBER	Not Null	PK—A unique, system-generated number to use as the record ID.
CREATION_DATE	DATE	Not Null	The date and time the row was created.
MODIFIED_DATE	DATE	Not Null	The date and time the row was last modified. This column's data is maintained using a database trigger.
RESOURCE_TYPE	VARCHAR(80)	Not Null	FK to L10N_RESOURCE_TYPE.

The L10N_LOCALE Database Table

This table defines the characteristics of a locale that are needed to localize an application.

Table 8-90 L10N_LOCALE Table Metadata

Column Name	Data Type	Null Value	Description
LOCALE_ID	NUMBER	Not Null	PK—A unique, system-generated number to use as the record ID.
CREATION_DATE	DATE	Not Null	The date and time the row was created.
MODIFIED_DATE	DATE	Not Null	The date and time the row was last modified. This column's data is maintained using a database trigger.
ENCODING	VARCHAR(20)	Not Null	The encoding that is used by the locale. The default encoding is UTF-8.
LANGUAGE	CHAR(2)	Not Null	Lowercase two-letter ISO-639 language code that is used by the locale; for example, en, au.
COUNTRY	CHAR(2)	Null	Uppercase two-letter ISO-3166 country code that is used by the locale; for example, US, UK.

Table 8-90 L10N_LOCALE Table Metadata (Continued)

Column Name	Data Type	Null Value	Description
VARIANT	VARCHAR(40)	Null	Vendor- and browser-specific code variant code that is used by the locale; for example, WIN, MAC, UNIX.

The L10N_RESOURCE Database Table

This table is used to define the localized title and description of a localized resource.

Table 8-91 L10N_RESOURCE Table Metadata

Column Name	Data Type	Null Value	Description
LOCALE_ID	NUMBER	Not Null	PK and FK to L10N_LOCALE.
INTERSECTION_ID	NUMBER	Not Null	PK and FK to L10N_INTERSECTION.
CREATION_DATE	DATE	Not Null	The date and time the row was created.
MODIFIED_DATE	DATE	Not Null	The date and time the row was last modified. This column's data is maintained using a database trigger.
TITLE	VARCHAR(80)	Not Null	A localized title for the object, typically used for display purposes; for example, the name of the portal or portlet.
DESCRIPTION	VARCHAR(500)	Null	A localized description of the object.

The L10N_RESOURCE_TYPE Database Table

This table is used to define portal resource types for localization.

Table 8-92 L10N_RESOURCE _TYPE Table Metadata

Column Name	Data Type	Null Value	Description
RESOURCE_TYPE	VARCHAR(80)	Not Null	PK—Type of resource to be localized; for example, BOOK, DESKTOP, DESKTOP CATEGORY.

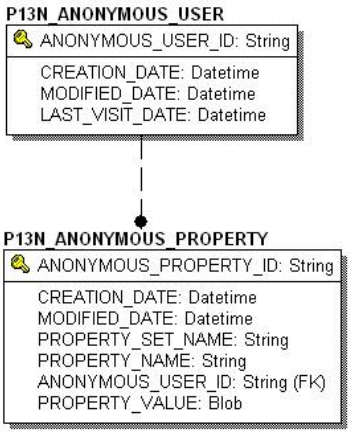
Table 8-92 L10N_RESOURCE _TYPE Table Metadata (Continued)

Column Name	Data Type	Null Value	Description
CREATION_DATE	DATE	Not Null	The date and time the row was created.
MODIFIED_DATE	DATE	Not Null	The date and time the row was last modified. This column's data is maintained using a database trigger.
APPLICATION_NAME	VARCHAR(100)	Not Null	The name of the application to which the resource belongs. APPLICATION_NAME is currently set to PORTAL for all types of resources to be localized.

Tracked Anonymous User Database Objects

This section documents the database objects for the WebLogic Portal package. Figure 8-15 shows the entity-relation diagram for the WebLogic Portal Anonymous User database objects.

Figure 8-15 Entity-Relation Diagram for the Anonymous User Tables



The Tracked Anonymous User Dictionary Tables

The following tables support tracking of anonymous users:

- The P13N_ANONYMOUS_PROPERTY Database Table

- The P13N_ANONYMOUS_USER Database Table

The P13N_ANONYMOUS_PROPERTY Database Table

This table is used to store the properties associated with the tracked anonymous user.

Table 8-93 P13N_ANONYMOUS_PROPERTY Table Metadata

Column Name	Data Type	Null Value	Description
ANONYMOUS_PROPERTY_ID	VARCHAR(128)	Not Null	PK—A unique, system-generated number to use as the record ID.
CREATION_DATE	DATE	Not Null	The date and time the row was created.
MODIFIED_DATE	DATE	Not Null	The date and time the row was last modified. This column's data is maintained using a database trigger.
PROPERTY_SET_NAME	VARCHAR(100)	Not Null	The name of the property set for which the tracked anonymous user data is set.
PROPERTY_NAME	VARCHAR(100)	Not Null	The name of the property being tracked for the anonymous user.
ANONYMOUS_USER_ID	VARCHAR(128)	Not Null	The foreign key that maps to the primary key of the anonymous user.
PROPERTY_VALUE	LONG RAW	Not Null	The value ". Must implement `java.io.Serializable`.

The P13N_ANONYMOUS_USER Database Table

This table is used to store the tracked anonymous user data.

Table 8-94 P13N_ANONYMOUS_USER Table Metadata

Column Name	Data Type	Null Value	Description
ANONYMOUS_USER_ID	VARCHAR(128)	Not Null	The foreign key that maps to the primary key of the anonymous user.
CREATION_DATE	DATE	Not Null	The date and time the row was created.

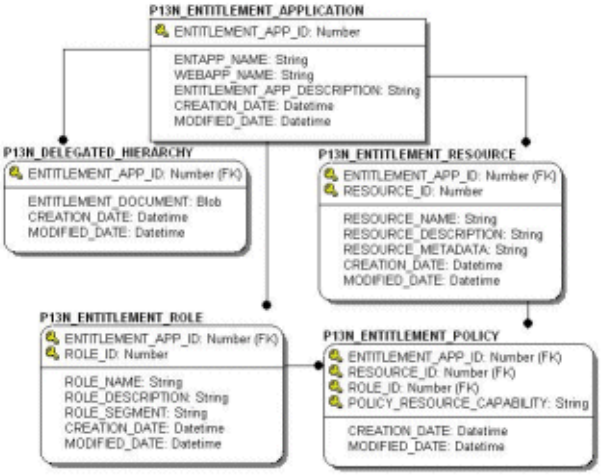
Table 8-94 P13N_ANONYMOUS_USER Table Metadata (Continued)

Column Name	Data Type	Null Value	Description
MODIFIED_DATE	DATE	Not Null	The date and time the row was last modified. This column's data is maintained using a database trigger.
LAST_VISIT_DATE	DATE	Null	Date when the tracked anonymous user last updated the data.

Entitlement Reference Database Objects

This section documents the database objects for the WebLogic Portal package. Figure 8-16 shows the entity-relation diagram for the WebLogic Portal Entitlement Reference database objects.

Figure 8-16 Entity-Relation Diagram for the Entitlement Reference Tables



The Entitlement Reference Dictionary Tables

The following tables are used by the Administration Portal to maintain security policy reference data as policies are created, edited, and deleted. These tables allow efficient searching for policies given a role name, resource ID web application name, and so on. The Entitlement Policy system has the following tables:

- The P13N_ENTITLEMENT_APPLICATION Database Table
- The P13N_ENTITLEMENT_POLICY Database Table
- The P13N_ENTITLEMENT_RESOURCE Database Table
- The P13N_ENTITLEMENT_ROLE Database Table
- The P13N_DELEGATED_HIERARCHY Database Table

The P13N_ENTITLEMENT_APPLICATION Database Table

This table is used to uniquely identify an application for which entitlements can be applied.

Table 8-95 P13N_ENTITLEMENT_APPLICATION Table Metadata

Column Name	Data Type	Null Value	Description
ENTITLEMENT_APP_ID	NUMBER(15)	Not Null	PK—A unique, system-generated number to use as the record ID.
ENTAPP_NAME	VARCHAR(255)	Not Null	The name of the enterprise application.
WEBAPP_NAME	VARCHAR(255)	Null	The name of the web application.
ENTITLEMENT_APP_DESCRIPTION	VARCHAR(255)	Null	The description of the enterprise application.
CREATION_DATE	DATE	Not Null	The date and time the record was created; the default is the current time stamp.
MODIFIED_DATE	DATE	Not Null	The date and time the record was last modified; the default is the current time stamp.

The P13N_ENTITLEMENT_POLICY Database Table

This table is used to uniquely identify an entitlement policy. An entitlement policy is created when an entitlement role is associated with an entitlement resource and capability.

Table 8-96 P13N_ENTITLEMENT_POLICY Table Metadata

Column Name	Data Type	Null Value	Description
RESOURCE_ENT_APP_ID	NUMBER(15)	Not Null	PK and FK to P13N_ENTITLEMENT_APPLICATION.
RESOURCE_ID	NUMBER(15)	Not Null	PK and FK to P13N_ENTITLEMENT_RESOURCE.
ROLE_ENT_APP_ID	NUMBER(15)	Not Null	PK and FK to P13N_ENTITLEMENT_APPLICATION.
ROLE_ID	NUMBER(15)	Not Null	PK and FK to P13N_ENTITLEMENT_ROLE.
POLICY_RESOURCE_CAPABILITY	VARCHAR(80)	Not Null	PK—Identifies the unique capability of this policy instance.
CREATION_DATE	DATE	Not Null	The date and time the record was created; the default is the current time stamp.
MODIFIED_DATE	DATE	Not Null	The date and time the record was last modified; the default is the current time stamp.

The P13N_ENTITLEMENT_RESOURCE Database Table

This table is used to uniquely identify an application resource that can have an entitlement associated with it.

Table 8-97 P13N_ENTITLEMENT_RESOURCE Table Metadata

Column Name	Data Type	Null Value	Description
ENTITLEMENT_APP_ID	NUMBER(15)	Not Null	PK and FK to P13N_ENTITLEMENT_APPLICATION.
RESOURCE_ID	NUMBER(15)	Not Null	PK—A unique, system-generated number to use as the record ID.
RESOURCE_NAME	VARCHAR(255)	Not Null	The name of the resource having a policy applied on it.

Table 8-97 P13N_ENTITLEMENT_RESOURCE Table Metadata (Continued)

Column Name	Data Type	Null Value	Description
RESOURCE_DESCRIPTION	VARCHAR(255)	Null	Optional description of resource.
RESOURCE_METADATA	VARCHAR(255)	Null	Optional application-defined metadata.
CREATION_DATE	DATE	Not Null	The date and time the record was last modified; the default is the current time stamp.
MODIFIED_DATE	DATE	Not Null	The date and time the record was last modified; the default is the current time stamp.

The P13N_ENTITLEMENT_ROLE Database Table

This table is used to uniquely identify entitlement and delegated administration roles for a given application.

Table 8-98 P13N_ENTITLEMENT_ROLE Table Metadata

Column Name	Data Type	Null Value	Description
ENTITLEMENT_APP_ID	NUMBER(15)	Not Null	PK and FK to P13N_ENTITLEMENT_APPLICATION.
ROLE_ID	NUMBER(15)	Not Null	PK—A unique, system-generated number to use as the record ID.
ROLE_NAME	VARCHAR(255)	Not Null	The name of the security role.
ROLE_DESCRIPTION	VARCHAR(255)	Null	Optional description of the role.
ROLE_SEGMENT	VARCHAR(255)	Null	Optional role expression name.
CREATION_DATE	DATE	Not Null	The date and time the record was last modified; the default is the current time stamp.
MODIFIED_DATE	DATE	Not Null	The date and time the record was last modified; the default is the current time stamp.

The P13N_DELEGATED_HIERARCHY Database Table

This table is used to uniquely identify an entitlement hierarchy. An entitlement hierarchy is associated with a P13N_ENTITLEMENT_APPLICATION.

Table 8-99 P13N_DELEGATED_HIERARCHY Table Metadata

Column Name	Data Type	Null Value	Description
ENTITLEMENT_APP_ID	NUMBER(15)	Not Null	PK and FK to P13N_ENTITLEMENT_APPLICATION.
ENTITLEMENT_DOCUMENT	CLOB	Not Null	An XML document containing the Delegated Administration role hierarchy.
CREATION_DATE	DATE	Not Null	The date and time the record was last modified; the default is the current time stamp.
MODIFIED_DATE	DATE	Not Null	The date and time the record was last modified; the default is the current time stamp.

WebLogic Portal DDL Modules

WebLogic Portal DDL Modules

WebLogic Portal Database Definition Language (DDL) modules are provided in directories with the following format:

WL_HOME/portal/db/*dbms_name*/*dbms_version*

For example:

WL_HOME/portal/db/pointbase/44

WL_HOME/portal/db/oracle/817

WL_HOME/portal/db/oracle/9i

Note: The same WebLogic Portal DDL is used for both Oracle 8.1.7 and 9i databases as indicated by a `readme.txt` file in the `oracle/9i` directory.This directory naming structure offers the ability to have distinct DDL between DBMS versions.

Data inserts for bootstrap data that must be inserted into tables in each WebLogic Portal database are contained in the following:

WL_HOME/portal/db/data/required/xx_insert_system_required_data.sql

WebLogic Portal DDL is provided in files named as follows:

xx_create_fkeys.sql

xx_create_indexes.sql

xx_create_tables.sql

xx_create_triggers.sql

```
xx_create_views.sql
xx_drop_constraints.sql
xx_drop_fkeys.sql
xx_drop_indexes.sql
xx_drop_tables.sql
xx_drop_views.sql
```

where the xx is a prefix from the table below:

Prefix	Description
au	Anonymous user
bt	Behavior Tracking
cm	Content Management
cmv	Content Management Versioning
collaboration*	Compoze portlets
ds	Data synchronization
er	Entitlement Reference
p13n	WebLogic Portal Personalization
pf	WebLogic Portal Framework and Localization
sample_cm	Content Management types data
wlcs	WebLogic Commerce Services
wps	WebLogic Portal Services
wsrp	Web Services for Remote Portlets

* Database object definitions for portlets from Compoze Software

APPENDIX B

Property Files and Database Scripts

The db_settings.properties file

Database scripts use the `db_settings.properties` file for these purposes: to connect to the database; to drop, create, or alter database objects; and for data inserts.

The `db_settings.properties` file exists in any domain directory that contains Portal, including Portal upgrade directories that contain a database upgrade script.

The database scripts (for example, `create_db.cmd` and `upgrade_db_schema_to_ 81SP4.sh`) use the `db_settings.properties` file that is located in the same directory from which you start the script.

Definitions Section

In the first section of the file the domain, modules, and object actions are defined. For more information on the Portal modules and their uses, see "WebLogic Portal DDL Modules" on page A-1.

Do not edit this section of the file. Removing modules is not recommended or supported, due to object dependencies.

```
domain_name=portalDomain
p13n_modules=p13n au bt ds er
portal_modules=cm cmv wlcs wps collaboration sample_cm
netuix_modules=pf wsrp
```

```
drop_actions=drop_views drop_fkeys drop_indexes drop_constraints
drop_tables

create_actions=create_tables create_fkeys create_indexes create_views
create_triggers
```

Database Parameters

Use the database parameters section of the file to define the database that you plan to use with WebLogic Portal. By default, the PointBase section is active. To use another database, place and remove comment characters as needed to activate the appropriate section of the file, and edit the parameter values for your environment.

For example, the PointBase and Oracle sections of the properties file are included below:

```
#------PointBase--------------------------------

#@IF_USING_POINTBASE@

database=POINTBASE

db_version=44

jdbcdriver=com.pointbase.jdbc.jdbcUniversalDriver

host=localhost

db_name=workshop

port=9093

dblogin=WEBLOGIC

dbpassword=WEBLOGIC

connection=jdbc:pointbase:server://localhost:9093/workshop

pointbase_ini=pointbase/pointbase.ini

#@ENDIF_USING_POINTBASE@

#

#------Oracle---------------------------------#

#

#@IF_USING_ORACLE@

#database=ORACLE

#db_version=817

#server=@ORACLE_NET_SERVICE_NAME@

#dblogin=@ORACLE_USER@
```

```
#dbpassword=@ORACLE_PASSWORD@
#@ENDIF_USING_ORACLE@
```

Scripts to Create or Upgrade Databases

WebLogic Portal database objects are created by running a `create_db.cmd/.sh` script from a domain directory or by using "Load Database" from the Configuration Wizard.

WebLogic Portal database upgrades for Service Packs are applied using one of the following scripts. See the Upgrade Guide for details on upgrading to a specific Service Pack.

Upgrading from 7.0 to 8.1 GA

WL_HOME`\portal\db\upgrade_db_schema_to_81.cmd/.sh`

Upgrading from 8.1 GA or SP2 to 8.1 SP3

WL_HOME`\portal\db\SP3\upgrade_db_schema_to_81SP3.cmd/.sh`

Upgrading from 8.1 SP3 to 8.1 SP4

WL_HOME`\portal\db\SP4\upgrade_db_schema_to_81SP4.cmd/.sh`

The scripts listed above call scripts residing in the *WL_HOME*`\portal\db` directory. With the exception of the default PointBase database, each database requires that a database client be installed and configured prior to script execution.

The following scripts located in *WL_HOME*`\portal\db` are called by the `create_db.cmd/.sh` scripts as well as the `upgrade_db_schema_to_<81`*version*`>` scripts. These scripts are not meant to be run independently. They are meant to be called by a `create_db.cmd/.sh` script or an `upgrade_db_schema_to_<81`*version*`>` script.

`create_db_common.cmd/sh`

This is a driver script that, depending on the database that is uncommented in `db_settings.properties`, calls other common and database-specific scripts.

`create_tmp_ddl.cmd/sh`

Called by `create_db_common.cmd/.sh` to read the various DDL modules (`.sql` files) for the appropriate database and database version, and to create `tmp*.sql` files (where *=Create, CreateUser, Drop, Insert) for building and populating database objects. Each module defined in `db_settings.properties` is looped through for files prefixed with a modules prefix and suffixed by a drop_actions, create_actions and alter_actions suffix to populate `tmp*.sql` files.

For a list of DDL modules, see "WebLogic Portal DDL Modules" on page A-1.

`create_<`*your_database*`>.cmd/sh`

where `<`*your_database*`>` has a value of: `pointbase`, `db2`, `ms_sql`, `oracle`, or `sybase`.

Connects to the database based on parameters set in `db_settings.properties`. The connection is performed using database client software. The `tmp*.sql` files created with `create_tmp_ddl.cmd/.sh` are processed to define database objects.

`db_version.properties`

Although this file has a type of `.properties`, it is not intended to be edited; the scripts use this file to identify the database version number directory from which the `.sql` DDL files are read. In the case of Oracle 8.1.7 and Oracle 9*i*, the same `.sql` files from the *`WL_HOME`*`\portal\db\oracle\817` directory are used for both database versions.

`create_stats`

Reads `.sql` scripts from *`WL_HOME`*`\portal\db\<`*`database`*`>\<`*`database_version`*`>\admin` and updates database optimizer statistics. Output is written to the file `statistics.log`. An installation report based on `.sql` scripts in the admin directory is also created. Statistics and the installation report are run for all databases except PointBase.

`load_data.cmd/sh`

This script is called by `create_db_common.cmd/.sh`, but it is not used to insert any data into database tables. If a `loadsampledata.properties` existed in the domain directory, this script would invoke the database loader and use its information to insert sample data into database tables.

Output from the `create_db.cmd/.sh` script is written to a `create_db.log` file in the directory from which the script is executed. Output from the `upgrade_db_schema_to_<81`*`version`*`>` scripts is written to `upgrade_db_schema_to_<81`*`version`*`>.log`.