

**Oracle® Fusion Middleware**

Database Administration Guide for Oracle WebLogic Portal

10g Release 3 (10.3.2)

**E14233-01**

February 2010

Copyright © 2010, Oracle and/or its affiliates. All rights reserved.

This software and related documentation are provided under a license agreement containing restrictions on use and disclosure and are protected by intellectual property laws. Except as expressly permitted in your license agreement or allowed by law, you may not use, copy, reproduce, translate, broadcast, modify, license, transmit, distribute, exhibit, perform, publish, or display any part, in any form, or by any means. Reverse engineering, disassembly, or decompilation of this software, unless required by law for interoperability, is prohibited.

The information contained herein is subject to change without notice and is not warranted to be error-free. If you find any errors, please report them to us in writing.

If this software or related documentation is delivered to the U.S. Government or anyone licensing it on behalf of the U.S. Government, the following notice is applicable:

U.S. GOVERNMENT RIGHTS Programs, software, databases, and related documentation and technical data delivered to U.S. Government customers are "commercial computer software" or "commercial technical data" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, the use, duplication, disclosure, modification, and adaptation shall be subject to the restrictions and license terms set forth in the applicable Government contract, and, to the extent applicable by the terms of the Government contract, the additional rights set forth in FAR 52.227-19, Commercial Computer Software License (December 2007). Oracle USA, Inc., 500 Oracle Parkway, Redwood City, CA 94065.

This software is developed for general use in a variety of information management applications. It is not developed or intended for use in any inherently dangerous applications, including applications which may create a risk of personal injury. If you use this software in dangerous applications, then you shall be responsible to take all appropriate fail-safe, backup, redundancy, and other measures to ensure the safe use of this software. Oracle Corporation and its affiliates disclaim any liability for any damages caused by use of this software in dangerous applications.

Oracle is a registered trademark of Oracle Corporation and/or its affiliates. Other names may be trademarks of their respective owners.

This software and documentation may provide access to or information on content, products, and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third-party content, products, and services. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third-party content, products, or services.

---

---

# Contents

|   |       |
|---|-------|
| <b>Preface</b> .....  | xvii  |
| Audience .....  | xvii  |
| Documentation Accessibility .....   | xvii  |
| Related Documents .....   | xviii |
| Conventions .....   | xviii |
| <br>  |       |
| <b>1 Introduction</b>   |       |
| 1.1 Introduction to Using Databases with WebLogic Portal .....                      | 1-1   |
| 1.2 How This Guide Fits into the Portal Life Cycle.....                             | 1-1   |
| <br>  |       |
| <b>2 Database Configuration Overview</b>  |       |
| 2.1 Overview of Enterprise-Quality Database Configuration for WebLogic Portal ..... | 2-1   |
| 2.2 Manually Creating Database Objects and JDBC Settings .....                      | 2-2   |
| 2.2.1 Creating Additional Content Management Databases .....                        | 2-2   |
| <br>  |       |
| <b>3 Database Setup and Maintenance Overview</b>                                    |       |
| 3.1 Encrypting Passwords .....  | 3-1   |
| 3.2 Choosing Internationalization and Localization Settings .....                   | 3-1   |
| 3.2.1 Changing the Language Settings in SQL Server .....                            | 3-2   |
| 3.2.2 Supporting Unicode Languages in SQL Server .....                              | 3-2   |
| 3.2.3 Choosing Character Sets and Sort Orders.....                                  | 3-3   |
| 3.2.3.1 Oracle .....  | 3-3   |
| 3.2.3.2 SQL Server .....  | 3-3   |
| 3.2.3.3 MySQL .....   | 3-3   |
| 3.2.3.4 Sybase.....   | 3-4   |
| 3.2.3.5 DB2.....  | 3-4   |
| 3.3 Performance Considerations .....  | 3-4   |
| 3.3.1 Collocating the Database and WebLogic Portal .....                            | 3-4   |
| 3.3.2 Using Caching to Reduce Database Access .....                                 | 3-4   |
| 3.3.3 Caching Database Tables Using Database-Specific Settings .....                | 3-5   |
| 3.3.4 Improving Concurrency Using Sybase Locking .....                              | 3-5   |
| 3.3.5 Improving Content Search Response Time for Oracle .....                       | 3-5   |
| 3.3.6 Using Database Statistics for Table and Index Organization.....               | 3-6   |
| 3.3.7 Optimizing Table Data and Index Placement .....                               | 3-6   |
| 3.3.7.1 Choosing Index Placement.....   | 3-6   |

|         |   |      |
|---------|---|------|
| 3.3.7.2 | Choosing Data Storage for Tables with CLOB/BLOB/TEXT/IMAGE Data ..... | 3-6  |
| 3.3.7.3 | Choosing Behavior Tracking Data Placement.....                        | 3-7  |
| 3.4     | Sizing Considerations.....  | 3-8  |
| 3.4.1   | Database Sizing .....   | 3-8  |
| 3.4.1.1 | Monitoring Behavior Tracking Tables.....                              | 3-8  |
| 3.4.1.2 | Monitoring Personalization Tables.....                                | 3-8  |
| 3.4.1.3 | Monitoring Portal Framework and WSRP Tables .....                     | 3-8  |
| 3.4.1.4 | Monitoring Content Management Database Tables.....                    | 3-9  |
| 3.4.1.5 | Monitoring Community Database Tables.....                             | 3-10 |
| 3.4.2   | Setting Page and Block Sizes .....                                    | 3-10 |
| 3.4.3   | Setting BLOB/CLOB Data Type Size Limits .....                         | 3-10 |
| 3.5     | Performing Database Backup and Recovery .....                         | 3-11 |
| 3.6     | Propagating Portal Environments .....                                 | 3-11 |
| 3.7     | Support for Asynchronous Proliferation Requires XA .....              | 3-11 |
| 3.8     | Note About Creating or Refreshing Database Objects .....              | 3-11 |

## 4 Using PointBase

|       |  |     |
|-------|--|-----|
| 4.1   | PointBase Documentation .....  | 4-1 |
| 4.2   | PointBase JAR Files.....   | 4-1 |
| 4.3   | PointBase Tools .....  | 4-2 |
| 4.4   | WebLogic Portal PointBase Databases.....                                   | 4-2 |
| 4.5   | PointBase Database Size Restriction.....                                   | 4-2 |
| 4.6   | Administering the WebLogic Portal PointBase Database.....                  | 4-2 |
| 4.6.1 | Launching the PointBase Console from the Windows Start Menu .....          | 4-3 |
| 4.6.2 | Launching the PointBase Console from the startPointBaseConsole Script..... | 4-3 |

## 5 Using Microsoft SQL Server

|     |  |     |
|-----|--|-----|
| 5.1 | Configuring Microsoft SQL Server Databases.....  | 5-1 |
| 5.2 | Configuring the Main WebLogic Portal Database .....                                    | 5-2 |
| 5.3 | Manually Creating a Separate Database and Database Objects for Behavior Tracking... .. | 5-3 |
| 5.4 | Database Scripts and Usage Notes .....   | 5-3 |

## 6 Using MySQL

|     |  |     |
|-----|--|-----|
| 6.1 | Configuring MySQL Databases .....  | 6-1 |
| 6.2 | Configuring the Main WebLogic Portal Database .....                                    | 6-2 |
| 6.3 | Manually Creating a Separate Database and Database Objects for Behavior Tracking... .. | 6-3 |
| 6.4 | Database Scripts and Usage Notes .....   | 6-4 |

## 7 Using Oracle

|       |  |     |
|-------|--|-----|
| 7.1   | Configuring Oracle Databases .....   | 7-1 |
| 7.2   | Configuring the Main WebLogic Portal Database .....                                    | 7-2 |
| 7.3   | Manually Creating a Separate Database and Database Objects for Behavior Tracking... .. | 7-3 |
| 7.4   | Database Scripts and Usage Notes .....   | 7-4 |
| 7.5   | Using WebLogic Portal with Oracle RAC .....  | 7-5 |
| 7.5.1 | Supported Configuration .....  | 7-5 |
| 7.5.2 | Configuration Considerations for Oracle RAC .....                                      | 7-5 |

|         |  |     |
|---------|--|-----|
| 7.5.2.1 | Multi-Data Source vs. Oracle Thin Driver .....           | 7-6 |
| 7.5.2.2 | Fail-over vs. Load-Balancing .....                       | 7-6 |
| 7.5.2.3 | XA Considerations .....                                  | 7-6 |
| 7.5.2.4 | Supported Configurations.....                            | 7-6 |
| 7.5.3   | Procedure for Using Oracle RAC with WebLogic Portal..... | 7-7 |
| 7.5.4   | Sample Configuration Files .....                         | 7-9 |

## 8 Using Sybase

|     |   |     |
|-----|---|-----|
| 8.1 | Configuring Sybase Databases.....   | 8-1 |
| 8.2 | Configuring the Main WebLogic Portal Database .....                                 | 8-2 |
| 8.3 | Manually Creating a Separate Database and Database Objects for Behavior Tracking... | 8-3 |
| 8.4 | Database Scripts and Usage Notes .....  | 8-4 |

## 9 Using DB2

|     |   |     |
|-----|---|-----|
| 9.1 | Configuring DB2 Databases .....   | 9-1 |
| 9.2 | Configuring the Main WebLogic Portal Database .....                                 | 9-2 |
| 9.3 | Manually Creating a Separate Database and Database Objects for Behavior Tracking... | 9-3 |
| 9.4 | Database Scripts and Usage Notes .....  | 9-4 |

## 10 Data Dictionary

|           |   |       |
|-----------|---|-------|
| 10.1      | Personalization Database Objects.....             | 10-2  |
| 10.1.1    | Base Personalization Database Tables .....        | 10-2  |
| 10.1.1.1  | USER_PROFILE Database Table .....                 | 10-3  |
| 10.1.1.2  | ENTITY Database Table .....                       | 10-3  |
| 10.1.1.3  | PROPERTY_KEY Database Table .....                 | 10-3  |
| 10.1.1.4  | PROPERTY_VALUE Database Table .....               | 10-4  |
| 10.1.1.5  | SEQUENCER Database Table.....                     | 10-4  |
| 10.1.1.6  | USERS Database Table.....                         | 10-5  |
| 10.1.1.7  | GROUPS Database Table.....                        | 10-5  |
| 10.1.1.8  | GROUPMEMBERS Database Table .....                 | 10-5  |
| 10.1.1.9  | P13N_QUIESCENT_STATE Database Table.....          | 10-6  |
| 10.1.1.10 | P13N_LEASE Database Table.....                    | 10-6  |
| 10.1.1.11 | Credential Service Dictionary Tables .....        | 10-7  |
| 10.1.1.12 | P13N_CREDENTIAL Database Table .....              | 10-7  |
| 10.1.1.13 | P13N_CREDENTIAL_APPLICATION Database Table .....  | 10-8  |
| 10.1.1.14 | P13N_CREDENTIAL_MASTER Database Table.....        | 10-8  |
| 10.1.1.15 | P13N_CREDENTIAL_METADATA Database Table.....      | 10-9  |
| 10.1.1.16 | Job Scheduler Dictionary Tables .....             | 10-10 |
| 10.1.1.17 | P13N_JOB_DEFINITION Database Table .....          | 10-10 |
| 10.1.1.18 | P13N_JOB_LISTENER Database Table .....            | 10-11 |
| 10.1.1.19 | P13N_JOB_PARAMETER Database Table .....           | 10-11 |
| 10.1.1.20 | P13N_JOB_RUN_HEADER Database Table .....          | 10-12 |
| 10.1.1.21 | P13N_JOB_RUN_DATA Database Table.....             | 10-13 |
| 10.1.1.22 | Entitlement Reference Dictionary Tables.....      | 10-13 |
| 10.1.1.23 | P13N_ENTITLEMENT_APPLICATION Database Table ..... | 10-13 |
| 10.1.1.24 | P13N_ENTITLEMENT_POLICY Database Table.....       | 10-14 |

|           |  |       |
|-----------|--|-------|
| 10.1.1.25 | P13N_ENTITLEMENT_RESOURCE Database Table .....     | 10-15 |
| 10.1.1.26 | P13N_ENTITLEMENT_ROLE Database Table.....          | 10-15 |
| 10.1.1.27 | P13N_DELEGATED_HIERARCHY Database Table .....      | 10-16 |
| 10.1.2    | Tracked Anonymous User Dictionary Tables .....     | 10-16 |
| 10.1.2.1  | P13N_ANONYMOUS_PROPERTY Database Table.....        | 10-16 |
| 10.1.2.2  | P13N_ANONYMOUS_USER Database Table .....           | 10-17 |
| 10.1.3    | Behavior Tracking Database Tables .....            | 10-17 |
| 10.1.3.1  | BT_EVENT_TYPE Database Table.....                  | 10-18 |
| 10.1.3.2  | BT_EVENT Database Table.....                       | 10-18 |
| 10.1.3.3  | BT_EVENT_ACTION Database Table .....               | 10-21 |
| 10.1.4    | Data Synchronization Database Tables .....         | 10-22 |
| 10.1.4.1  | DATA_SYNC_APPLICATION Database Table .....         | 10-22 |
| 10.1.4.2  | DATA_SYNC_ITEM Database Table .....                | 10-22 |
| 10.1.4.3  | DATA_SYNC_SCHEMA_URI Database Table.....           | 10-23 |
| 10.1.4.4  | DATA_SYNC_VERSION Database Table .....             | 10-23 |
| 10.2      | Portal Services Database Objects .....             | 10-24 |
| 10.2.1    | AD_BUCKET Database Table .....                     | 10-24 |
| 10.2.2    | AD_COUNT Database Table .....                      | 10-25 |
| 10.2.3    | PLACEHOLDER_PREVIEW Database Table.....            | 10-25 |
| 10.2.4    | MAIL_ADDRESS Database Table .....                  | 10-25 |
| 10.2.5    | MAIL_BATCH Database Table .....                    | 10-25 |
| 10.2.6    | MAIL_BATCH_ENTRY Database Table .....              | 10-26 |
| 10.2.7    | MAIL_HEADER Database Table .....                   | 10-26 |
| 10.2.8    | MAIL_MESSAGE Database Table .....                  | 10-26 |
| 10.2.9    | SCENARIO_END_STATE Database Table .....            | 10-26 |
| 10.3      | Portal Framework Database Objects .....            | 10-27 |
| 10.3.1    | Portal Framework Database Tables .....             | 10-27 |
| 10.3.1.1  | PF_BOOK_DEFINITION Database Table.....             | 10-28 |
| 10.3.1.2  | PF_BOOK_GROUP Database Table .....                 | 10-29 |
| 10.3.1.3  | PF_BOOK_INSTANCE Database Table.....               | 10-29 |
| 10.3.1.4  | PF_DESKTOP_DEFINITION Database Table.....          | 10-30 |
| 10.3.1.5  | PF_DESKTOP_INSTANCE Database Table.....            | 10-32 |
| 10.3.1.6  | PF_LAYOUT_DEFINITION Database Table .....          | 10-32 |
| 10.3.1.7  | PF_LOOK_AND_FEEL_DEFINITION Database Table.....    | 10-33 |
| 10.3.1.8  | PF_MARKUP_DEFINITION Database Table .....          | 10-34 |
| 10.3.1.9  | PF_MARKUP_XML Database Table.....                  | 10-35 |
| 10.3.1.10 | PF_MENU_DEFINITION Database Table.....             | 10-35 |
| 10.3.1.11 | PF_PAGE_DEFINITION Database Table .....            | 10-36 |
| 10.3.1.12 | PF_PAGE_INSTANCE Database Table .....              | 10-37 |
| 10.3.1.13 | PF_PLACEHOLDER_DEFINITION Database Table .....     | 10-37 |
| 10.3.1.14 | PF_PLACEMENT Database Table .....                  | 10-38 |
| 10.3.1.15 | PF_PORTAL Database Table .....                     | 10-38 |
| 10.3.1.16 | PF_PORTLET_CATEGORY Database Table .....           | 10-39 |
| 10.3.1.17 | PF_PORTLET_CATEGORY_DEFINITION Database Table..... | 10-39 |
| 10.3.1.18 | PF_PORTLET_DEFINITION Database Table .....         | 10-40 |
| 10.3.1.19 | PF_PORTLET_INSTANCE Database Table .....           | 10-41 |
| 10.3.1.20 | PF_PORTLET_PREFERENCE Database Table.....          | 10-42 |

|           |   |       |
|-----------|---|-------|
| 10.3.1.21 | PF_PORTLET_PREFERENCE_VALUE Database Table.....                     | 10-43 |
| 10.3.1.22 | PF_PORTLET_USER_PROPERTIES Database Table .....                     | 10-43 |
| 10.3.1.23 | PF_SHELL_DEFINITION Database Table.....                             | 10-44 |
| 10.3.1.24 | PF_SHELL_PLACEMENT Database Table .....                             | 10-45 |
| 10.3.1.25 | PF_THEME_DEFINITION Database Table .....                            | 10-46 |
| 10.3.1.26 | PF_URL_COMPRESSION Database Table.....                              | 10-47 |
| 10.3.2    | Localization Dictionary Tables .....                                | 10-47 |
| 10.3.2.1  | L10N_INTERSECTION Database Table .....                              | 10-47 |
| 10.3.2.2  | L10N_LOCALE Database Table.....                                     | 10-47 |
| 10.3.2.3  | L10N_RESOURCE Database Table.....                                   | 10-48 |
| 10.3.2.4  | L10N_RESOURCE_TYPE Database Table .....                             | 10-49 |
| 10.4      | WSRP Portal Framework Database Objects.....                         | 10-49 |
| 10.4.1    | PF_CONSUMER_PORTLETS Database Table .....                           | 10-49 |
| 10.4.2    | PF_CONSUMER_PROPERTIES Database Table .....                         | 10-49 |
| 10.4.3    | PF_CONSUMER_REGISTRY Database Table.....                            | 10-50 |
| 10.4.4    | PF_PRODUCER_REGISTRY Database Table.....                            | 10-51 |
| 10.4.5    | PF_PROXY_PORTLET_INSTANCE Database Table .....                      | 10-52 |
| 10.4.6    | PF_PROXY_BOOK Database Table .....                                  | 10-53 |
| 10.4.7    | PF_PROXY_PAGE Database Table.....                                   | 10-54 |
| 10.5      | Communities Framework Database Objects.....                         | 10-55 |
| 10.5.1    | Communities Portal Framework Data Dictionary Tables .....           | 10-55 |
| 10.5.1.1  | PF_COMMUNITY_DEFINITION Database Table.....                         | 10-56 |
| 10.5.1.2  | PF_COMMUNITY_MEMBER Database Table .....                            | 10-57 |
| 10.5.1.3  | PF_COMMUNITY_MEMBERSHIP Database Table .....                        | 10-58 |
| 10.5.1.4  | PF_MEMBERSHIP_CAPABILITY Database Table.....                        | 10-59 |
| 10.5.1.5  | PF_COMMUNITY_PROPERTY Database Table .....                          | 10-59 |
| 10.5.1.6  | PF_COMMUNITY_PROPERTY_VALUE Database Table .....                    | 10-60 |
| 10.5.1.7  | PF_FILE_TEMPLATE Database Table .....                               | 10-60 |
| 10.5.1.8  | PF_INVITATION Database Table.....                                   | 10-61 |
| 10.5.1.9  | PF_INVITEE Database Table .....                                     | 10-62 |
| 10.5.1.10 | PF_INVITEE_PROPERTY Database Table .....                            | 10-63 |
| 10.5.1.11 | PF_MESSAGING_ADDR Database Table .....                              | 10-63 |
| 10.5.1.12 | PF_NOTIFICATION Database Table .....                                | 10-63 |
| 10.5.1.13 | PF_NOTIFICATION_PAYLOAD Database Table .....                        | 10-64 |
| 10.6      | Content Management Database Objects.....                            | 10-66 |
| 10.6.1    | Content Management Data Dictionary Tables .....                     | 10-66 |
| 10.6.1.1  | CM_NODE Database Table .....  | 10-66 |
| 10.6.1.2  | CM_COLLABORATION_API_METADATA Database Table .....                  | 10-68 |
| 10.6.1.3  | CM_OBJECT_CLASS Database Table .....                                | 10-69 |
| 10.6.1.4  | CM_PROPERTY Database Table.....                                     | 10-70 |
| 10.6.1.5  | CM_PROPERTY_CHOICE Database Table.....                              | 10-72 |
| 10.6.1.6  | CM_PROPERTY_DEFINITION Database Table.....                          | 10-74 |
| 10.6.1.7  | CM_PROPERTY_TEXT Database Table.....                                | 10-76 |
| 10.6.1.8  | CM_WORKFLOW Database Table.....                                     | 10-76 |
| 10.7      | Content Management Virtual Content Repository Database Objects..... | 10-77 |
| 10.7.1    | Content Management Virtual Object Data Dictionary Tables .....      | 10-77 |
| 10.7.1.1  | CMV_NODE Table.....   | 10-77 |

|          |                                      |       |
|----------|--------------------------------------|-------|
| 10.7.1.2 | CMV_NODE_ASSIGNED_ROLE Table.....    | 10-78 |
| 10.7.1.3 | CMV_NODE_VERSION Table.....          | 10-78 |
| 10.7.1.4 | CMV_PROPERTY Table.....              | 10-78 |
| 10.7.1.5 | CMV_VALUE Table.....                 | 10-79 |
| 10.7.1.6 | CMV_VALUE_TEXT Table.....            | 10-81 |
| 10.7.1.7 | CMV_NODE_VERSION_PROPERTY Table..... | 10-81 |
| 10.8     | RDBMS Security Store Tables .....    | 10-82 |

## **A Scripts and Properties Files**

|       |  |     |
|-------|--|-----|
| A.1   | WebLogic Portal DDL Modules.....                                       | A-1 |
| A.2   | Content Management DDL Modules.....                                    | A-2 |
| A.3   | Personalization DDL Modules.....                                       | A-3 |
| A.4   | Using the database.properties File .....                               | A-3 |
| A.4.1 | Setting the Database Parameters in the Properties File.....            | A-4 |
| A.5   | Scripts to Create an Additional Content Management Repository .....    | A-5 |
| A.6   | Scripts to Manually Upgrade from Version 8.1, 9.2, 10.0, or 10.2 ..... | A-5 |
| A.7   | Scripts to Drop Deprecated Compoze Database Tables .....               | A-5 |
| A.8   | Scripts to Drop Deprecated RDBMS Authenticator Tables .....            | A-5 |

## **B Database Schema Diagrams**





## List of Figures

|      |   |     |
|------|---|-----|
| B-1  | Personalization Database Tables .....                               | B-2 |
| B-2  | Personalization Credential Service Database Tables .....            | B-2 |
| B-3  | Personalization Job Scheduler Database Tables .....                 | B-3 |
| B-4  | Entitlement Database Tables .....                                   | B-3 |
| B-5  | Behavior Tracking Database Tables .....                             | B-4 |
| B-6  | Data Synchronization Database Tables .....                          | B-4 |
| B-7  | Portal Services Database Tables.....                                | B-5 |
| B-8  | Portal Framework Database Tables.....                               | B-5 |
| B-9  | WSRP Portal Framework Tables.....                                   | B-6 |
| B-10 | Communities Portal Framework Database Tables.....                   | B-7 |
| B-11 | Content Management Database Tables .....                            | B-8 |
| B-12 | Content Management Virtual Content Repository Database Tables ..... | B-8 |

## List of Examples

|     |                                       |      |
|-----|---------------------------------------|------|
| 7-1 | p13nDataSource Multi-Data Source..... | 7-9  |
| 7-2 | p13nDataSource-1 Data Source.....     | 7-10 |
| 7-3 | p13nDataSource-2 Data Source.....     | 7-10 |



## List of Tables

|       |   |       |
|-------|---|-------|
| 5-1   | Database Scripts and Usage Notes .....                  | 5-4   |
| 6-1   | Database Scripts and Usage Notes .....                  | 6-4   |
| 7-1   | Database Scripts and Usage Notes .....                  | 7-4   |
| 7-2   | WebLogic Portal Data Sources When Using Oracle RAC..... | 7-6   |
| 8-1   | Database Scripts and Usage Notes .....                  | 8-5   |
| 9-1   | Database Scripts and Usage Notes .....                  | 9-5   |
| 10-1  | USER_PROFILE Table Metadata .....                       | 10-3  |
| 10-2  | ENTITY Table Metadata .....                             | 10-3  |
| 10-3  | PROPERTY_KEY Table Metadata .....                       | 10-4  |
| 10-4  | PROPERTY_VALUE Table Metadata .....                     | 10-4  |
| 10-5  | SEQUENCER Table Metadata .....                          | 10-5  |
| 10-6  | USERS Table Metadata.....                               | 10-5  |
| 10-7  | GROUPS Table Metadata .....                             | 10-5  |
| 10-8  | GROUPMEMBERS Table Metadata.....                        | 10-5  |
| 10-9  | P13N_QUIESCENT_STATE Table Metadata .....               | 10-6  |
| 10-10 | P13N_LEASE Table Metadata.....                          | 10-7  |
| 10-11 | P13N_CREDENTIAL Table Metadata .....                    | 10-7  |
| 10-12 | P13N_CREDENTIAL_APPLICATION Table Metadata .....        | 10-8  |
| 10-13 | P13N_CREDENTIAL_MASTER Table Metadata .....             | 10-9  |
| 10-14 | P13N_CREDENTIAL_METADATA Table Metadata .....           | 10-9  |
| 10-15 | P13N_JOB_DEFINITION Table Metadata .....                | 10-10 |
| 10-16 | P13N_JOB_LISTENER Table Metadata .....                  | 10-11 |
| 10-17 | P13N_JOB_PARAMETER Table Metadata .....                 | 10-12 |
| 10-18 | P13N_JOB_RUN_HEADER Table Metadata .....                | 10-12 |
| 10-19 | P13N_JOB_RUN_DATA Table Metadata .....                  | 10-13 |
| 10-20 | P13N_ENTITLEMENT_APPLICATION Table Metadata .....       | 10-14 |
| 10-21 | P13N_ENTITLEMENT_POLICY Table Metadata .....            | 10-14 |
| 10-22 | P13N_ENTITLEMENT_RESOURCE Table Metadata.....           | 10-15 |
| 10-23 | P13N_ENTITLEMENT_ROLE Table Metadata.....               | 10-15 |
| 10-24 | P13N_DELEGATED_HIERARCHY Table Metadata .....           | 10-16 |
| 10-25 | P13N_ANONYMOUS_PROPERTY Table Metadata.....             | 10-17 |
| 10-26 | P13N_ANONYMOUS_USER Table Metadata .....                | 10-17 |
| 10-27 | BT_EVENT_TYPE Database Table .....                      | 10-18 |
| 10-28 | BT_EVENT Database Table .....                           | 10-18 |
| 10-29 | XML_DEFINITION Data Elements.....                       | 10-19 |
| 10-30 | BT_EVENT_ACTION Table Metadata .....                    | 10-21 |
| 10-31 | DATA_SYNC_APPLICATION Table Metadata .....              | 10-22 |
| 10-32 | DATA_SYNC_ITEM Table Metadata .....                     | 10-22 |
| 10-33 | DATA_SYNC_SCHEMA_URI Table Metadata.....                | 10-23 |
| 10-34 | DATA_SYNC_VERSION Table Metadata .....                  | 10-23 |
| 10-35 | AD_BUCKET Table Metadata.....                           | 10-24 |
| 10-36 | AD_COUNT Table Metadata.....                            | 10-25 |
| 10-37 | PLACEHOLDER_PREVIEW Table Metadata .....                | 10-25 |
| 10-38 | MAIL_ADDRESS Table Metadata.....                        | 10-25 |
| 10-39 | MAIL_BATCH Table Metadata .....                         | 10-26 |
| 10-40 | MAIL_BATCH_ENTRY Table Metadata.....                    | 10-26 |
| 10-41 | MAIL_HEADER Table Metadata .....                        | 10-26 |
| 10-42 | MAIL_MESSAGE Table Metadata .....                       | 10-26 |
| 10-43 | SCENARIO_END_STATE Table Metadata.....                  | 10-27 |
| 10-44 | PF_BOOK_DEFINITION Table Metadata.....                  | 10-28 |
| 10-45 | PF_BOOK_GROUP Table Metadata .....                      | 10-29 |
| 10-46 | PF_BOOK_INSTANCE Table Metadata.....                    | 10-30 |
| 10-47 | PF_DESKTOP_DEFINITION Table Metadata.....               | 10-30 |

|        |  |       |
|--------|--|-------|
| 10-48  | PF_DESKTOP_INSTANCE Table Metadata .....           | 10-32 |
| 10-49  | PF_LAYOUT_DEFINITION Table Metadata .....          | 10-32 |
| 10-50  | PF_LOOK_AND_FEEL_DEFINITION Table Metadata.....    | 10-34 |
| 10-51  | PF_MARKUP_DEFINITION Table Metadata .....          | 10-35 |
| 10-52  | PF_MARKUP_XML Table Metadata .....                 | 10-35 |
| 10-53  | PF_MENU_DEFINITION Table Metadata.....             | 10-36 |
| 10-54  | PF_PAGE_DEFINITION Table Metadata .....            | 10-36 |
| 10-55  | PF_PAGE_INSTANCE Table Metadata .....              | 10-37 |
| 10-56  | PF_PLACEHOLDER_DEFINITION Table Metadata.....      | 10-38 |
| 10-57  | PF_PLACEMENT Table Metadata .....                  | 10-38 |
| 10-58  | PF_PORTAL Table Metadata .....                     | 10-39 |
| 10-59  | PF_PORTLET_CATEGORY Table Metadata .....           | 10-39 |
| 10-60  | PF_PORTLET_CATEGORY_DEFINITION Table Metadata..... | 10-39 |
| 10-61  | PF_PORTLET_DEFINITION Table Metadata .....         | 10-40 |
| 10-62  | PF_PORTLET_INSTANCE Table Metadata .....           | 10-42 |
| 10-63  | PF_PORTLET_PREFERENCE Table Metadata .....         | 10-43 |
| 10-64  | PF_PORTLET_PREFERENCE_VALUE Table Metadata .....   | 10-43 |
| 10-65  | PF_PORTLET_USER_PROPERTIES Table Metadata.....     | 10-44 |
| 10-66  | PF_SHELL_DEFINITION Table Metadata.....            | 10-44 |
| 10-67  | PF_SHELL_PLACEMENT Table Metadata .....            | 10-45 |
| 10-68  | PF_THEME_DEFINITION Table Metadata.....            | 10-46 |
| 10-69  | PF_URL_COMPRESSION Table Metadata.....             | 10-47 |
| 10-70  | L10N_INTERSECTION Table Metadata .....             | 10-47 |
| 10-71  | L10N_LOCALE Table Metadata.....                    | 10-48 |
| 10-72  | L10N_RESOURCE Table Metadata.....                  | 10-48 |
| 10-73  | L10N_RESOURCE_TYPE Table Metadata.....             | 10-49 |
| 10-74  | PF_CONSUMER_PORTLETS Database Table .....          | 10-49 |
| 10-75  | PF_CONSUMER_PROPERTIES Database Table .....        | 10-50 |
| 10-76  | PF_CONSUMER_REGISTRY Database Table.....           | 10-50 |
| 10-77  | PF_PRODUCER_REGISTRY Database Table .....          | 10-51 |
| 10-78  | PF_PROXY_PORTLET_INSTANCE Database Table .....     | 10-52 |
| 10-79  | PF_PROXY_BOOK Table Metadata .....                 | 10-54 |
| 10-80  | PF_PROXY_PAGE Table Metadata.....                  | 10-54 |
| 10-81  | PF_COMMUNITY_DEFINITION Database Table .....       | 10-56 |
| 10-82  | PF_COMMUNITY_MEMBER Database Table.....            | 10-58 |
| 10-83  | PF_COMMUNITY_MEMBERSHIP Database Table.....        | 10-58 |
| 10-84  | PF_COMMUNITY_MEMBERSHIP Database Table.....        | 10-59 |
| 10-85  | PF_COMMUNITY_PROPERTY Database Table.....          | 10-60 |
| 10-86  | PF_COMMUNITY_PROPERTY_VALUE Database Table.....    | 10-60 |
| 10-87  | PF_FILE_TEMPLATE Database Table.....               | 10-61 |
| 10-88  | PF_INVITATION Database Table .....                 | 10-61 |
| 10-89  | PF_INVITEE Database Table.....                     | 10-62 |
| 10-90  | PF_INVITEE_PROPERTY Database Table.....            | 10-63 |
| 10-91  | PF_NOTIFICATION Database Table.....                | 10-64 |
| 10-92  | PF_NOTIFICATION_PAYLOAD Database Table.....        | 10-65 |
| 10-93  | CM_NODE Table Metadata .....                       | 10-67 |
| 10-94  | CM_COLLABORATION_API_METADATA Table Metadata ..... | 10-68 |
| 10-95  | CM_OBJECT_CLASS Table Metadata .....               | 10-69 |
| 10-96  | CM_PROPERTY Table Metadata .....                   | 10-70 |
| 10-97  | CM_PROPERTY_CHOICE Table Metadata.....             | 10-73 |
| 10-98  | CM_PROPERTY_DEFINITION Table Metadata .....        | 10-75 |
| 10-99  | CM_PROPERTY_TEXT Table Metadata .....              | 10-76 |
| 10-100 | CM_WORKFLOW Table.....                             | 10-77 |
| 10-101 | CMV_NODE Table .....                               | 10-78 |
| 10-102 | CMV_NODE_ASSIGNED_ROLE Table.....                  | 10-78 |

|        |   |       |
|--------|---|-------|
| 10-103 | CMV_NODE_VERSION Table .....                      | 10-78 |
| 10-104 | CMV_PROPERTY Table Metadata.....                  | 10-79 |
| 10-105 | CMV_VALUE Table .....                             | 10-79 |
| 10-106 | CMV_VALUE_TEXT Table Metadata .....               | 10-81 |
| 10-107 | CMV_NODE_VERSION_PROPERTY Table .....             | 10-82 |
| 10-108 | RDBMS Security Store Database Table Defaults..... | 10-82 |
| A-1    | WebLogic Portal DDL Module File Prefixes.....     | A-2   |
| A-2    | Content Management DDL Module File Prefixes.....  | A-2   |
| A-3    | Personalization DDL Module File Prefixes.....     | A-3   |





---

---

# Preface

This guide describes how to configure and manage databases for Oracle WebLogic Portal. The guide also contains a data dictionary and information about Data Definition Language (DDL) modules, which are the SQL scripts that create WebLogic Portal database objects.

## Audience

This document is intended for portal developers and database administrators.

## Documentation Accessibility

Our goal is to make Oracle products, services, and supporting documentation accessible to all users, including users that are disabled. To that end, our documentation includes features that make information available to users of assistive technology. This documentation is available in HTML format, and contains markup to facilitate access by the disabled community. Accessibility standards will continue to evolve over time, and Oracle is actively engaged with other market-leading technology vendors to address technical obstacles so that our documentation can be accessible to all of our customers. For more information, visit the Oracle Accessibility Program Web site at <http://www.oracle.com/accessibility/>.

### Accessibility of Code Examples in Documentation

Screen readers may not always correctly read the code examples in this document. The conventions for writing code require that closing braces should appear on an otherwise empty line; however, some screen readers may not always read a line of text that consists solely of a bracket or brace.

### Accessibility of Links to External Web Sites in Documentation

This documentation may contain links to Web sites of other companies or organizations that Oracle does not own or control. Oracle neither evaluates nor makes any representations regarding the accessibility of these Web sites.

### Deaf/Hard of Hearing Access to Oracle Support Services

To reach Oracle Support Services, use a telecommunications relay service (TRS) to call Oracle Support at 1.800.223.1711. An Oracle Support Services engineer will handle technical issues and provide customer support according to the Oracle service request process. Information about TRS is available at <http://www.fcc.gov/cgb/consumerfacts/trs.html>, and a list of phone numbers is available at <http://www.fcc.gov/cgb/dro/trsphonebk.html>.

## Related Documents

For more information, see the following documents in the WebLogic Portal documentation set:

- Oracle WebLogic Portal Portal Development Guide
- *Oracle WebLogic Portal Production Operations Guide*

## Conventions

The following text conventions are used in this document:

| <b>Convention</b> | <b>Meaning</b>   |
|-------------------|--|
| <b>boldface</b>   | Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.         |
| <i>italic</i>     | Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.                          |
| monospace         | Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter. |

---

---

# Introduction

This guide describes how to configure and manage databases for Oracle WebLogic Portal. The guide also contains a data dictionary and information about Data Definition Language (DDL) modules, which are the SQL scripts that create WebLogic Portal database objects.

For detailed information about 9.2 database upgrade tasks, see the *Oracle Fusion Middleware Upgrade Guide for Oracle WebLogic Portal*.

This chapter includes the following sections:

- [Section 1.1, "Introduction to Using Databases with WebLogic Portal"](#)
- [Section 1.2, "How This Guide Fits into the Portal Life Cycle"](#)

## 1.1 Introduction to Using Databases with WebLogic Portal

The WebLogic family of products uses Java Database Connectivity (JDBC) to interact with the databases that store program and application data. JDBC is a standard Java API that consists of classes and interfaces written in the Java programming language. Application, tool, and database developers use JDBC to write database applications and execute SQL statements.

WebLogic JDBC enables programmers to interact with different database management systems (DBMSs) including Oracle, Microsoft SQL Server, MySQL, Sybase, and IBM DB2. WebLogic Portal executes DDL commands for each DBMS it interacts with, so you must ensure that your configuration is supported by reviewing the document "Oracle Fusion Middleware Supported System Configurations."

You can perform basic JDBC setup using the WebLogic Configuration Wizard. The Configuration Wizard allows you to set up JDBC connection pools and data sources for WebLogic Portal. The wizard is described in *Oracle Fusion Middleware Creating Domains Using the Configuration Wizard*.

You can use the WebLogic Server Administration Console to enable, configure, and monitor JDBC connection pools, data sources, and MultiPools. For detailed information on these tasks, see *Oracle Fusion Middleware Configuring and Managing JDBC for Oracle WebLogic Server*.

The remaining chapters of this book describe database management tasks and reference information that is specific to WebLogic Portal.

## 1.2 How This Guide Fits into the Portal Life Cycle

In general, the tasks that you perform to plan your WebLogic Portal database structure occur as part of the architecture phase of the portal life cycle; these tasks are described

in the *Oracle Fusion Middleware Overview for Oracle WebLogic Portal*. The tasks that you perform to manage your WebLogic Portal database structure can occur during any other phase of the portal life cycle; for this reason, this book is not organized according to the portal life cycle structure.

---

---

## Database Configuration Overview

This chapter describes how to set up and start using a database management system (DBMS) with WebLogic Portal.

The sample data provided with WebLogic Portal uses the PointBase DBMS. PointBase is a demonstration database ships with WebLogic Server to allow you to use sample user accounts and data. For more information about PointBase, see [Chapter 4, "Using PointBase."](#)

---

---

**Note:** PointBase is supported only for the design, development, and verification of applications; it is not supported for enterprise-quality deployment. The evaluation license of PointBase has a database size limit of 30 MB.

---

---

To maximize the performance of your database configuration, see [Section 3.3, "Performance Considerations."](#)

This chapter contains the following sections:

- [Section 2.1, "Overview of Enterprise-Quality Database Configuration for WebLogic Portal"](#)
- [Section 2.2, "Manually Creating Database Objects and JDBC Settings"](#)
- [Section 2.2.1, "Creating Additional Content Management Databases"](#)

### 2.1 Overview of Enterprise-Quality Database Configuration for WebLogic Portal

There are several steps to configuring a database for use with WebLogic Portal. You can perform these tasks manually using provided sample scripts, or you can use the WebLogic Configuration Wizard for certain steps.

To configure databases for use with WebLogic Portal, follow these steps:

1. Review related documentation to ensure that your configuration is supported in "Oracle Fusion Middleware Supported System Configurations."
2. Create your database or database instance. If you want to use behavior event tracking in your enterprise-quality environment, consider using a separate database for behavior event tracking. For more information, see the chapter in this guide pertaining to your database vendor.
3. Prepare the database for use with WebLogic Portal. Oracle provides sample initialization scripts that you must modify and run on the vendor database before

using the database with WebLogic Portal. For more information about these initialization scripts and how to use them, see the chapter in this guide that pertains to your database vendor.

4. Create appropriate database objects and set JDBC driver settings, either using the Configuration Wizard or by modifying and running the sample scripts.

To perform this step using the Configuration Wizard, see *Oracle Fusion Middleware Creating Domains Using the Configuration Wizard*.

To perform this step manually, see [Section 2.2, "Manually Creating Database Objects and JDBC Settings,"](#) next.

5. Optionally, manually create any additional content management repositories.

## 2.2 Manually Creating Database Objects and JDBC Settings

In some cases, you might want to manually create database objects and JDBC settings rather than using the WebLogic Configuration Wizard.

The following are reasons you might want to perform database configuration manually:

- To create additional content management repositories.
- To set up your enterprise-quality (non-PointBase) database.
- If your target database was not configured using the WebLogic Configuration Wizard.
- If, after running the Configuration Wizard, you decide that you want your domain to point to a different database.
- To create a subset of WebLogic Portal database objects; for example, you might want to create only behavior tracking database objects for a particular database.
- To refresh your database with the base configuration data that comes with the product.

---

---

**Caution:** Oracle's database creation scripts first drop all database objects and then they recreate them, which means that all data added since your original installation will be lost. After running the database configuration scripts, only the base configuration data that is needed for the product will exist.

---

---

For instructions on manually configuring your database, see the chapter in this guide pertaining to your database vendor.

### 2.2.1 Creating Additional Content Management Databases

You need to manually create any additional content repositories. Instructions for doing this are in the *Oracle Fusion Middleware Content Management Guide for Oracle WebLogic Portal*.

---

---

## Database Setup and Maintenance Overview

This section identifies considerations for setting up and maintaining WebLogic Portal databases.

For a discussion of database environment considerations for team development, see the *Oracle Fusion Middleware Production Operations Guide for Oracle WebLogic Portal*.

For additional information on vendor-specific considerations and recommendations, see the chapter in this guide pertaining to your database vendor.

This chapter includes the following sections:

- [Section 3.1, "Encrypting Passwords"](#)
- [Section 3.2, "Choosing Internationalization and Localization Settings"](#)
- [Section 3.3, "Performance Considerations"](#)
- [Section 3.4, "Sizing Considerations"](#)
- [Section 3.5, "Performing Database Backup and Recovery"](#)
- [Section 3.6, "Propagating Portal Environments"](#)
- [Section 3.7, "Support for Asynchronous Proliferation Requires XA"](#)
- [Section 3.8, "Note About Creating or Refreshing Database Objects"](#)

### 3.1 Encrypting Passwords

WebLogic Portal's database creation scripts (`create_db.cmd/sh`) use the domain's salt file when they call `JDBCDataLoader`.

Database passwords in the database properties files, such as `database.properties`, can be either plain text or encrypted. Use `weblogic.security.Encrypt` to obtain the encrypted passwords. The salt file used to encrypt and decrypt passwords is the domain's `DOMAIN_HOME/security/SerializedSystemIni.dat` salt file.

For more about `weblogic.security.Encrypt`, see "Using the WebLogic Server Java Utilities" in the *Oracle Fusion Middleware Command Reference for Oracle WebLogic Server*.

### 3.2 Choosing Internationalization and Localization Settings

The Localization Industry Standards Association (LISA) defines internationalization and localization as follows:

"Internationalization is the process of generalizing a product so that it can handle multiple languages and cultural conventions without the need for re-design. Internationalization takes place at the level of program design and document development."

"Localization involves taking a product and making it linguistically and culturally appropriate to the target locale (country/region and language) where it will be used and sold."

This section contains information on choosing language settings, character sets, and collation (sort) settings for your database. It includes the sections:

- [Section 3.2.1, "Changing the Language Settings in SQL Server"](#)
- [Section 3.2.2, "Supporting Unicode Languages in SQL Server"](#)
- [Section 3.2.3, "Choosing Character Sets and Sort Orders"](#)

### 3.2.1 Changing the Language Settings in SQL Server

To use SQL Server with portal applications that are localized, you must modify the language settings in the <WLPORAL\_HOME>\portal\db\sql\_server\admin\create\_database.sql script before running it to create the database.

For example, to change the language setting from US English to Japanese, perform the following edits:

1. Change the line reading:

```
COLLATE SQL_Latin1_General_CP1_CS_AS
```

to:

```
COLLATE Japanese_CS_AS_KS_WS
```

2. Change the line reading:

```
exec sp_addlogin 'WEBLOGIC', 'WEBLOGIC', 'WEBLOGIC', 'us_english'
```

to:

```
exec sp_addlogin 'WEBLOGIC', 'WEBLOGIC', 'WEBLOGIC', 'japanese'
```

3. Run the script (after making any additional required modifications).

For more information about language support in SQL Server, refer to the vendor documentation.

### 3.2.2 Supporting Unicode Languages in SQL Server

If you want to localize portal applications to support a unicode language, you can edit and run the <WLPORAL\_HOME>\portal\db\sql\_server\admin\alter\_use\_unicode\_columns.sql script to alter all columns of type TEXT, CHAR, and VARCHAR to use their unicode equivalents (NTEXT, NCHAR, and NVARCHAR, respectively).

---

---

**Caution:** Changing these column data types has performance implications. Each column that is modified uses twice as much storage space as the original data type. Back up the database that contains the portal schema before running this script.

---

---



After running the script, you can use `DBCC CHECKDB` command to verify physical storage and table constraints.

### 3.2.3 Choosing Character Sets and Sort Orders

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

For a globalized WebLogic Portal application, choose the database's character set and 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.

The following sections describe character set and sort order considerations for WebLogic Portal-supported databases. Refer to your vendor documentation for more information.

#### 3.2.3.1 Oracle

For Oracle, you define the character set when you create the database. In globalized Oracle 9i and 10g databases, a commonly used character set is `AL32UTF8`.

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

#### 3.2.3.2 SQL Server

For SQL Server, you can define both instance-level and database-level character sets and sort orders.

The default instance-level character set for 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 `<WLPORTAL_HOME>\portal\db\sql_server\admin\create_database.sql`. It defines the WebLogic Portal database with the setting `COLLATE SQL_Latin1_General_CP1_CS_AS`.

This specifies a case-sensitive definition. Setting the database to be case-sensitive allows content management queries to function for SQL Server as they do for other databases. If content management search queries are not written for a case-sensitive environment, you may have unexpected results.

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

#### 3.2.3.3 MySQL

For MySQL, you define the character set and set collate at database creation time. A common MySQL character set and collate are `CHARACTER SET latin1` and `COLLATE latin1_general_cs`.

MySQL restricts key lengths to 1000 bytes when a UTF8 character set is used. If UTF8 is the desired character set, the `CM_NODE.FULL_PATH` and `CM_PROPERTY.PROPERTY_NAME` columns are restricted to a maximum size of `VARCHAR(255)`, so by default database object creation errors will be thrown if the

MySQL database character set is UTF8. Therefore, do not use the UTF8 character set with MySQL at this time.

#### **3.2.3.4 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.

#### **3.2.3.5 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 use the DB2 `CODESET` configuration parameter to determine a database's codeset.

## **3.3 Performance Considerations**

This section details factors to consider when optimizing database performance. It includes the sections:

- [Section 3.3.1, "Collocating the Database and WebLogic Portal"](#)
- [Section 3.3.2, "Using Caching to Reduce Database Access"](#)
- [Section 3.3.3, "Caching Database Tables Using Database-Specific Settings"](#)
- [Section 3.3.4, "Improving Concurrency Using Sybase Locking"](#)
- [Section 3.3.5, "Improving Content Search Response Time for Oracle"](#)
- [Section 3.3.6, "Using Database Statistics for Table and Index Organization"](#)
- [Section 3.3.7, "Optimizing Table Data and Index Placement"](#)

### **3.3.1 Collocating the Database and WebLogic Portal**

To prevent network latency issues, you should locate your database and your WebLogic Portal instances in the same data center. This recommendation is especially important when you use the Propagation Utility, to ensure that a large propagation process can complete successfully.

### **3.3.2 Using Caching to Reduce Database Access**

The frequency of database access varies greatly depending on what functionality is used and how it is used. By reducing the frequency of database access, caching plays a large role in configuring the portal for optimum performance.

For the core portal framework, the portal taxonomy that all users share is loaded in memory during initialization, so database access is typically infrequent after initialization. If a user has customized the portal, a database read is necessary the first time they access it, to retrieve their customizations, user preferences, and so on.

Many other features of the product also use the database, including the content management store. If content is frequently accessed, caching is a good idea where possible; otherwise, database access is necessary to retrieve the content.

### 3.3.3 Caching Database Tables Using Database-Specific Settings

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 summarizes the support provided by each database type for caching or pinning database tables. Depending on your environment, you might 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 the `DBCC PINTABLE (database_id, table_id)` command to pin tables to the buffer cache.
- Sybase provides the ability to define multiple database caches and to selectively bind tables, indexes, and logs to those caches. Use `sp_objects_stats` to identify 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 the `BUFFERPOOL` configuration parameter to cache database data; this should be the focus of tuning efforts. DB2 allows you to assign individual tables and indexes to a buffer pool.

### 3.3.4 Improving Concurrency Using 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.

The following WebLogic tables are defined with `LOCK DATAROWS` for Sybase. Other WebLogic Portal tables for Sybase 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 the setting `LOCK DATAROWS`.

### 3.3.5 Improving Content Search Response Time for Oracle

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

To improve response time, verify that the `optimizer_mode` Oracle initialization parameter is set to `choose`.

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

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

Refer to your Oracle documentation for the more information about these settings, and the impact modifications may have on your installation.

### 3.3.6 Using Database Statistics for Table and Index Organization

Each DBMS has its own utility or commands for updating the database statistics used by its query optimizer. The statistics that you collect for your database often provide information on the organization of tables and indexes in your database. A DBA should schedule periodic jobs to maintain database statistics.

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

### 3.3.7 Optimizing Table Data and Index Placement

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

#### 3.3.7.1 Choosing Index Placement

For Oracle databases, use the `rebuild_indexes.sql` script to move indexes into their own tablespace, such as the `WEBLOGIC_INDEX` tablespace. Run this script manually after creating the WebLogic Portal database to move indexes into their own tablespace. For more information, see [Section 7.1, "Configuring Oracle Databases."](#)

For SQL Server and Sybase, 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. The provided sample `create_database.sql` scripts define `WEBLOGIC_INDEX` appropriately.

#### 3.3.7.2 Choosing 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 WebLogic Portal database platforms, CLOB/BLOB/TEXT/IMAGE table data should be physically stored separately from other types of data. However, to simplify deployment and to allow for flexibility in your configuration, the default WebLogic 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 configuration** - 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 relevant table columns are:

- CMV\_VALUE.BLOB\_VALUE
- CM\_PROPERTY.BLOB\_VALUE
- CM\_PROPERTY\_CHOICE.BLOB\_VALUE

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

The relevant table column is BT\_EVENT.XML\_DEFINITION.

- **Disc controller configuration** - If your environment has a separate disk controller and drive that can be dedicated to CLOB/BLOB/TEXT/IMAGE storage, you might see performance improvements using a separate controller rather than sharing a controller.
- **RAID storage configuration** - If you are using RAID, you may see limited 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 vendor documentation for specific details on changing settings for CLOB/BLOB/TEXT/IMAGE data.

The following additional columns 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

### 3.3.7.3 Choosing Behavior Tracking Data Placement

Behavior tracking enables you to develop, manage, and measure personalized portal applications. By recording events, you can customize portal applications and track visitor behavior.

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 other than PointBase, see the chapter in this guide pertaining to your database vendor for information on creating a separate database for behavior tracking events.

## 3.4 Sizing Considerations

This section describes the considerations for database, page, block, and data type sizing.

### 3.4.1 Database Sizing

Sizing 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 portlets such as My Mail Portlet, My Task List Portlet, and so on?
  - Can end users create their own portal resource views?

---

---

**Note:** WebLogic 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.

#### 3.4.1.1 Monitoring Behavior Tracking Tables

The BT\_EVENT table can grow significantly if behavior tracking is enabled.

#### 3.4.1.2 Monitoring Personalization Tables

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

- ENTITY – This table grows as you define new users and groups for your WebLogic Portal application.
- PROPERTY\_VALUE – This table grows 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 *Oracle Fusion Middleware User Management Guide for Oracle WebLogic Portal*.

#### 3.4.1.3 Monitoring Portal Framework and WSRP Tables

The following tables involve the Portal Framework and 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 is 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:

$$\text{number of users} * \text{number of pages} * \text{number of portlets} = \text{number of rows}$$

- 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, known as the primary instance. Every time a portlet is dragged onto a page, a new instance of the portlet is created and a column is inserted into this 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," another instance is created and a row inserted. (Subsequent changes to the portlet instance do not create a new instance/row.) This growth can be expressed using the equation:

$$\text{number of portlet definitions} * \text{number of instances} = \text{number of rows}$$

- 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

$$\text{number of portals} * \text{number of desktops} + \text{number of visitors who customize} = \text{number of rows}$$

#### 3.4.1.4 Monitoring Content Management Database Tables

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

- CM\_PROPERTY – This table stores property values. This table can grow significantly if content management is used to store large quantities of data.
- CM\_NODE – This table uniquely identifies a node within a WLP repository. 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 WLP repository (from the CM\_NODE table) that has been versioned and is being edited within the Virtual Content 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 Virtual Content Repository. This table can grow significantly if content management versioning is enabled (the default).

- **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 contains cross-references.
- **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 (the default).
- **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 is likely be the largest content management table because it stores the metadata associated with each content node.

#### 3.4.1.5 Monitoring Community Database Tables

The following database tables store Community data; you should monitor them for data growth. If you are using multiple Communities or have a large numbers of users, you can expect these tables to consume more space than the basic Portal Framework customization tables.

- **PF\_COMMUNITY\_DEFINITION**, **PF\_COMMUNITY\_PROPERTY** and **PF\_INVITATION** scale in proportion to the number of Community instances.
- **PF\_COMMUNITY\_MEMBER** scales in proportion to the number of WebLogic Server users involved in Communities.
- **PF\_COMMUNITY\_MEMBERSHIP**, **PF\_MEMBERSHIP\_CAPABILITY**, **PF\_INVITEE**, **PF\_INVITEE\_PROPERTY** and **PF\_NOTIFICATION** scale in proportion to the number of Community instances multiplied by the number of Community members per Community.

### 3.4.2 Setting Page and Block Sizes

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

For DB2 databases, an 8K buffer pool 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 page size is required for several WebLogic Portal tables and indexes. For Sybase the default page size is 2K, and Sybase does not allow rows to span pages. 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.

---

---

**Note:** See the chapter in this guide pertaining to your database vendor for additional details on database-specific considerations.

---

---

### 3.4.3 Setting BLOB/CLOB Data Type Size Limits

When you create database tables for DB2 and PointBase databases, the maximum BLOB/CLOB size varies by column. You can edit the appropriate database creation script to change this value.



In general, you should be aware of any size restrictions that might apply to your database. Restrictions can vary from one database version to another; refer to your vendor documentation for more information.

### 3.5 Performing Database Backup and Recovery

Use the same procedures for backup and recovery of WebLogic Portal databases as you use for other database data.

The following are general recommendations for backup and recovery:

- Store your data and J2EE resources in source control, and back up the source control database.
- Back up your WebLogic Portal database according to your database vendor's recommendations.

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

### 3.6 Propagating Portal Environments

You use the WebLogic Portal Propagation Utility to move portal metadata from one portal environment to another. You should be aware of the usage of this utility, and back up your databases prior to propagation.

By design, when you move a portal environment, not all database data is propagated. For example, user and group data is not propagated.

For more information about the Propagation Utility, and what database data is propagated, see the *Oracle Fusion Middleware Production Operations Guide for Oracle WebLogic Portal*. This guide also describes database environment considerations for team development.

### 3.7 Support for Asynchronous Proliferation Requires XA

JMS and JMS database tables are created automatically by WebLogic Server if they do not already exist in the database to store proliferation information. The Portal Resource Proliferation Mode setting determines how updates to portal resources (pages, books, portlets) are cascaded - or proliferated - to desktops and user-customized instances. You can set the mode to Asynchronous, Synchronous, or Off in the WebLogic Portal Administration Console.

When proliferation is performed asynchronously (the default) a JMS queue is used, which requires an XA driver. In this case, the `portalDataSourceAlwaysXA.xml` data source is used. This means that your database environment must be configured to support XA.

### 3.8 Note About Creating or Refreshing Database Objects

To create or refresh your WebLogic Portal database:

1. Stop WebLogic Server.
2. Delete the contents of the directory `DOMAIN_HOME/servers/yourServer/data/ldap`.
3. Run the `create_db.cmd` (or `create_db.sh`) command.



---

---

## Using PointBase

The sample data provided with WebLogic Portal uses the PointBase DBMS. PointBase is a demonstration database ships with WebLogic Server to allow you to use sample user accounts and data. PointBase is the default database when you create a domain with the Configuration Wizard.

---

---

**Note:** PointBase is supported only for the design, development, and verification of applications; it is not supported for enterprise-quality deployment. The evaluation license of PointBase has a database size limit of 30 MB.

---

---

The PointBase server is started automatically when you start WebLogic Server; it must be running for your applications to access it.

This chapter contains the following sections:

- [Section 4.1, "PointBase Documentation"](#)
- [Section 4.2, "PointBase JAR Files"](#)
- [Section 4.3, "PointBase Tools"](#)
- [Section 4.4, "WebLogic Portal PointBase Databases"](#)
- [Section 4.5, "PointBase Database Size Restriction"](#)
- [Section 4.6, "Administering the WebLogic Portal PointBase Database"](#)

### 4.1 PointBase Documentation

PointBase documentation for the version of PointBase supported by WebLogic Server is distributed with WebLogic Server in Adobe PDF form in the `<WEBLOGIC_HOME>\common\eval\pointbase\docs\embedded\pdf` directory. The PointBase documentation consists of the following manuals:

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

### 4.2 PointBase JAR Files

Refer to the section titled "PointBase JAR Files" in the "Before You Begin" chapter of the PointBase System Guide for information on the JAR files provided with WebLogic Server in the `<WEBLOGIC_HOME>\common\eval\pointbase\lib` directory.

## 4.3 PointBase Tools

Scripts for starting the PointBase Server and the PointBase console are distributed with WebLogic Server in the `<WEBLOGIC_HOME>\common\eval\pointbase\tools` directory. Scripts are called by start scripts in the sample domains and by 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.

## 4.4 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 `weblogic_eval.dbn` and log files for WebLogic Portal are named `weblogic_eval$1.wal`. Pre-built PointBase data, log, and `PointBase.ini` files for WebLogic Portal samples are included in the following directory:

```
<WLPORTAL_HOME>\samples\domains\portal
```

---

---

**Note:** To see the samples directory, you must install the Portal Examples using the Installation Wizard. See "Installing the Portal Examples" in the *Oracle Fusion Middleware Release Notes for Oracle WebLogic Portal* for details.

---

---

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

```
<MW_HOME>\user_projects\domains\DomainName
```

## 4.5 PointBase Database Size Restriction

The evaluation license of PointBase has a database size limit of 30 MB. You might find that you reach this limit if you use PointBase extensively. If this happens, you can upgrade your PointBase license by contacting [sales@pointbase.com](mailto:sales@pointbase.com), or reduce the amount of data by deleting table rows and running the Database Compress Tool to compress the database to a more compact size. You must shut down the server before running this command line tool; no other process can access the database while the tool runs. For more information, see the PointBase Developer's Guide (`<WEBLOGIC_HOME>\common\eval\pointbase\docs\embedded\pdf\pbdeveloper.pdf`).

## 4.6 Administering the WebLogic Portal PointBase Database

You can administer PointBase using the PointBase administration 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. This starts the PointBase Server.

This section contains the following sections:

- [Section 4.6.1, "Launching the PointBase Console from the Windows Start Menu"](#)

- [Section 4.6.2, "Launching the PointBase Console from the startPointBaseConsole Script"](#)

## 4.6.1 Launching the PointBase Console from the Windows Start Menu

To launch the PointBase Console from the Windows Start menu:

1. Go to **Start > All Programs > Oracle Products > Examples > WebLogic Portal > PointBase Console**.

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

2. 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 for a portal domain you create using the Configuration Wizard:**  
jdbc:pointbase:server://localhost:9093/weblogic\_eval
- **URL for sample portal domain:**  
jdbc:pointbase:server://localhost:9052/weblogic\_eval
- **User:** weblogic
- **Password:** weblogic

## 4.6.2 Launching the PointBase Console from the startPointBaseConsole Script

To launch the PointBase Console using the script:

---



---

**Note:** To see the samples directory, you must install the Portal Examples using the Installation Wizard. See "Installing the Portal Examples" in the *Oracle Fusion Middleware Release Notes for Oracle WebLogic Portal* for details.

---



---

1. Change directories to <WLPORTAL\_HOME>\samples\domains\portal\bin.
2. For a domain created by the Configuration Wizard, navigate to that domain's home directory.
3. Execute the appropriate start script—startPointBaseConsole.cmd or startPointBaseConsole.sh—to launch the PointBase Console.
4. 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 for a portal domain you create using the Configuration Wizard:**  
jdbc:pointbase:server://localhost:9093/weblogic\_eval
  - **URL for sample portal domain:**  
jdbc:pointbase:server://localhost:9052/weblogic\_eval
  - **User:** weblogic

- **Password:** weblogic

---

---

## Using Microsoft SQL Server

This chapter describes the steps necessary to use a Microsoft SQL Server database with WebLogic Portal, and includes the following sections:

- [Section 5.1, "Configuring Microsoft SQL Server Databases"](#)
- [Section 5.2, "Configuring the Main WebLogic Portal Database"](#)
- [Section 5.3, "Manually Creating a Separate Database and Database Objects for Behavior Tracking"](#)
- [Section 5.4, "Database Scripts and Usage Notes"](#)

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

### 5.1 Configuring Microsoft SQL Server Databases

Before proceeding, read [Section 2.1, "Overview of Enterprise-Quality Database Configuration for WebLogic Portal."](#)

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

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). Any additional content management repositories require a separate database user.

---

---

**Note:** The SQL Server JDBC driver requires additional steps to configure stored procedures for JTA and to place a required DLL on the SQL Server database host. For instructions, see "Installing Stored Procedures for JTA" in *Oracle Fusion Middleware Type 4 JDBC Drivers for Oracle WebLogic Server*.

---

---

To configure a SQL Server database:

1. Be sure to back up your database before installing any new database objects. See your database vendor documentation for details.
2. Review the provided sample scripts, located in the `<WLPORTAL_HOME>\portal\db\sql_server\admin` directory. See [Table 5-1](#) and the comments in the scripts for additional information.

3. Copy and modify the sample scripts appropriately for your environment to create each of the following databases:
  - a. Follow the steps in [Section 5.2, "Configuring the Main WebLogic Portal Database"](#) to create the main WebLogic Portal database and database objects.
  - b. If you want to create a separate behavior tracking database, follow the steps in [Section 5.3, "Manually Creating a Separate Database and Database Objects for Behavior Tracking."](#)

## 5.2 Configuring the Main WebLogic Portal Database

To configure the main WebLogic Portal database, follow these steps:

1. Copy the `create_database.sql` script and modify it appropriately for your environment. See [Table 5-1](#) and the comments in the script for additional information.

2. Run the modified `create_database.sql` script as a user with System Administrator privileges (normally the `sa` user). For example, from `osql`:

```
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.

Follow the remaining steps only if you want to create database objects manually rather than using the Configuration Wizard. To perform the remaining steps using the Configuration Wizard, see *Oracle Fusion Middleware Creating Domains Using the Configuration Wizard*.

1. Open your domain's `database.properties` file for edit.
2. Set `database=sql_server`.
3. Update the following settings (by replacing the `@` symbols and the text between the symbols with the correct values) for your main WebLogic Portal database:

```
sql_server.user=@DB_USER@
sql_server.password=@DB_PASSWORD@
sql_server.url=jdbc:bea:sqlserver://@DB_HOST@:@DB_PORT@;DatabaseName=@DB_NAME@
```

4. Create the database objects.
  - a. Navigate to the `<MW_HOME>\user_projects\domains\myPortalDomain` directory.
  - b. Double-click `create_db.cmd`.

If any error messages are displayed, check the `create_db.log` file for additional information.

---

**Note:** After running the `create_db` script, you need to update the database administrator password. See [Section 3.8, "Note About Creating or Refreshing Database Objects"](#) for detailed information.

---

5. Replace the JDBC data sources in your domain, which point to PointBase by default, with data sources that point to SQL Server. You can configure them using the WebLogic Server Administration Console or choose from the samples provided and update them for your database environment. Sample `jdbc.xml` definition files for each database and driver that Oracle supports are available in



the <WLPORTAL\_HOME>\portal\db\jdbc\database\_driver directory; for example, sql\_server\_bea. Follow the instructions in the <WLPORTAL\_HOME>\portal\db\jdbc\README.txt file.

## 5.3 Manually Creating a Separate Database and Database Objects for Behavior Tracking

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 "Setting Up Events and Behavior Tracking" in the *Oracle Fusion Middleware Interaction Management Guide for Oracle WebLogic Portal*.

---

**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.

---

To create a separate database for behavior tracking:

1. Modify the `bt_create_database.sql` file for your environment, as indicated in the instructions contained in the scripts and in [Table 5-1](#).
2. Run `bt_create_database.sql` as a user with system administrator privileges. For example, from `osql`:

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

If any error messages are displayed, check the `bt_create_database.log` file for additional information.

3. Navigate to the appropriate database directory based on your environment; for example, <WLPORTAL\_HOME>\p13n\db\sql\_server.
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_triggers.sql`
5. Run the following scripts from this directory: <WLPORTAL\_HOME>\portal\db\data\required:
  - `bt_insert_system_data.sql`
  - `bt9_insert_system_data.sql`
6. Use the WebLogic Server Administration Console to configure a non-XA JDBC data source to access your behavior tracking database. Associate the JNDI name `p13n.trackingDataSource` with that data source and then remove `p13n.trackingDataSource` from `p13nDataSource`.

## 5.4 Database Scripts and Usage Notes

[Table 5-1](#) describes the scripts that enable you to configure a Microsoft SQL Server database.

**Table 5–1 Database Scripts and Usage Notes**

| Script Name            | Description  |
|------------------------|--|
| create_database.sql    | <p>Creates a database. You must create a main WebLogic Portal database.</p> <p>Make a copy of this script and edit it to replace &lt;&lt;WEBLOGIC&gt;&gt; with the appropriate database name, database owner user, and password for each database you create. You must also edit the script to reflect valid disk locations for DATA devices, LOG devices, and the WEBLOGIC_INDEX file group; you may also need to modify file sizes. Put DATA and LOG files on separate physical disks and away from any system database files, unless you are using RAID devices.</p> <p><b>Note:</b> Do not change the name of the WEBLOGIC_INDEX file group.</p> <p>Typical names for the main WebLogic Portal database are:</p> <ul style="list-style-type: none"> <li>■ Database name: WEBLOGIC</li> <li>■ Database owner user: WEBLOGIC</li> <li>■ Password: WEBLOGIC</li> </ul> <p>When you run the script with these values, it 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. It also sets the WEBLOGIC database as the default database for the WEBLOGIC user.</p> <p><b>Note:</b> If you decide to create an additional content management repository, you must create a database with different user names for it. For additional information, see the <i>Oracle Fusion Middleware Content Management Guide for Oracle WebLogic Portal</i>.</p> |
| statistics.sql         | <p>Runs <code>sp_updatestats</code> to compute database statistics needed for the database optimizer. Update statistics periodically and whenever any significant changes in database data occur. (This is done by default for SQL Server databases with the <code>AUTO_UPDATE_STATISTICS</code> database option enabled.)</p> <p>When set to ON (the default), existing statistics are automatically updated when the data in the tables has changed.</p> <p>When set to OFF, existing statistics are not automatically updated. You must manually update statistics.</p> <p>The <code>AUTO_UPDATE_STATISTICS</code> option setting is set in the <code>IsAutoUpdateStatistics</code> property of the <code>DATABASEPROPERTYEX</code> function.</p>   |
| install_report.sql     | Builds an informational installation report about the database objects created in the WEBLOGIC schema.   |
| bt_create_database.sql | <p>Creates 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.</p> <p>You must edit the script to reflect valid disk locations for the DATA and LOG devices, or to modify file sizes. Put DATA and LOG files on separate physical disks and away from any system database files.</p> <p>The default names are the following:</p> <ul style="list-style-type: none"> <li>■ Database name: WEBLOGIC_EVENT</li> <li>■ Database owner user: WEBLOGIC_EVENT</li> <li>■ Password: WEBLOGIC_EVENT</li> </ul>  |

---

---

## Using MySQL

This chapter describes the steps necessary to use a MySQL 5.0 database with WebLogic Portal, and includes the following sections:

- [Section 6.1, "Configuring MySQL Databases"](#)
- [Section 6.2, "Configuring the Main WebLogic Portal Database"](#)
- [Section 6.3, "Manually Creating a Separate Database and Database Objects for Behavior Tracking"](#)
- [Section 6.4, "Database Scripts and Usage Notes"](#)

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

### 6.1 Configuring MySQL Databases

Before proceeding, read [Section 2.1, "Overview of Enterprise-Quality Database Configuration for WebLogic Portal."](#)

---

---

**Caution:** The MySQL JDBC driver is not fully XA compatible, therefore WebLogic Portal does not support using the `com.mysql.jdbc.jdbc2.optional.MysqlXADataSource`. Data sources that require XA and those that are XA capable should use the `com.mysql.jdbc.Driver` with the `global-transactions-protocol` `LoggingLastResource` instead.

---

---

The database creation scripts install domain-specific tables. You should work with a database administrator to modify the sample scripts, and to create the database users and tablespaces needed for your environment.

Multiple database users are required if you have multiple domains, or to run multiple environments using the same MySQL instance (for example, if you want to run development and system test from a single MySQL installation). Any additional content management repositories require a separate database user.

To configure a MySQL database:

1. Back up your database environment before installing any new database objects. See your database documentation for details.
2. Review the provided sample scripts, located in the `<WLPORTAL_HOME>/portal/db/mysql/admin` directory. See [Section 6.4, "Database Scripts and Usage Notes"](#) and the comments in the scripts for additional information.

3. Copy and modify the sample scripts appropriately for your environment to create each of the following database schemas:
  - a. Follow the steps in [Section 6.2, "Configuring the Main WebLogic Portal Database"](#) to create the main WebLogic Portal user and database objects.
  - b. If you want to create a separate behavior tracking database, follow the steps in [Section 6.3, "Manually Creating a Separate Database and Database Objects for Behavior Tracking."](#)

## 6.2 Configuring the Main WebLogic Portal Database

Perform the following steps to configure the main WebLogic Portal database:

1. Copy the `create_database.sql` script and modify it appropriately for your environment. See [Section 6.4, "Database Scripts and Usage Notes"](#) and the comments in the script for additional information. For more information on character sets, see [Section 3.2.3, "Choosing Character Sets and Sort Orders."](#)
2. Run the modified `create_database.sql` script as a user with user account creation privileges (normally the root user). For example, from `mysql`:

```
mysql -uroot --password=password <create_database.sql
```

Follow the remaining steps only if you want to create database objects manually rather than using the Configuration Wizard. To perform the remaining steps using the Configuration Wizard, see *Oracle Fusion Middleware Creating Domains Using the Configuration Wizard*.

1. Open your domain's `database.properties` file for edit.
2. Set `database=mysql`.
3. Update the following settings (by replacing the @ symbols and the text between the symbols with the correct values) for your main WebLogic Portal database:

```
mysql.user=@DB_USER@
mysql.password=@DB_PASSWORD@
mysql.url=jdbc:bea:mysql://@DB_HOST@:@DB_PORT@;SID=@DB_NAME@
```

4. Create the database objects.
  - a. Navigate to the `<MW_HOME>\user_projects\domains\myPortalDomain` directory.
  - b. Enter one of the commands:
    - On Windows, double-click `create_db.cmd`.
    - On UNIX, run `create_db.sh`.

If any error messages are displayed, check the `create_db.log` file for additional information.

---



---

**Note:** After running `create_db` script, you need to update the database administrator password. See [Section 3.8, "Note About Creating or Refreshing Database Objects"](#) for detailed information.

---



---

5. Replace the JDBC data sources in your domain, which point to PointBase by default, with data sources that point to MySQL. You can configure them using the WebLogic Server Administration Console or choose from the samples provided

and update them for your database environment. Sample `jdbc.xml` definition files for MySQL are available in the `<WLPORTAL_HOME>\portal\db\jdbc\mysql` directory. Follow the instructions in the `<WLPORTAL_HOME>\portal\db\jdbc\README.txt` file.

---

**Note:** Do this while WebLogic Server is stopped.

---

## 6.3 Manually Creating a Separate Database and Database Objects for Behavior Tracking

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 "Setting Up Events and Behavior Tracking" in the *Oracle Fusion Middleware Interaction Management Guide for Oracle WebLogic Portal*.

---

**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.

---

To create a separate database for behavior tracking:

1. Modify the `bt_create_database.sql` file for your environment, as indicated in the instructions contained in the scripts and in [Section 6.4, "Database Scripts and Usage Notes."](#)
2. Run `bt_create_database.sql` as a user with system administrator privileges. For example, from `osql`:
 

```
mysql -uroot --password=password <bt_create_database.sql
```
3. Navigate to the appropriate database directory based on your environment:
 

```
<WLPORTAL_HOME>\portal\db\mysql
```
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_triggers.sql`
5. Run the following scripts from the path `<WLPORTAL_HOME>\portal\db\data\required`:
  - `bt_insert_system_data.sql`
  - `bt9_insert_system_data.sql`
6. Use the WebLogic Server Administration Console to configure a non-XA JDBC data source to access your behavior tracking database. Associate the JNDI name `p13n.trackingDataSource` with that data source and then remove `p13n.trackingDataSource` from `p13nDataSource`.

## 6.4 Database Scripts and Usage Notes

Table 6–1 describes the scripts that enable you to configure the MySQL database and create WebLogic Portal objects in that database.

**Table 6–1 Database Scripts and Usage Notes**

| Script Name            | Description  |
|------------------------|--|
| create_database.sql    | <p>Creates a database. You must create a main WebLogic Portal database.</p> <p>Make a copy of this script and edit it to replace &lt;&lt;WEBLOGIC&gt;&gt; with the appropriate database name, database owner user, and password for each database you create.</p> <p>Typical names for the main WebLogic Portal database are:</p> <ul style="list-style-type: none"> <li>■ Database name: WEBLOGIC</li> <li>■ Database owner user: WEBLOGIC</li> <li>■ Password: WEBLOGIC</li> </ul> <p>When you run the script with these values, it creates the WEBLOGIC database and the WEBLOGIC database owner user, and grants privileges to the WEBLOGIC user in order to create database objects.</p> <p><b>Note:</b> If you decide to create an additional content management repository, you must create a database with different user names for it. For additional information, see the <i>Oracle Fusion Middleware Content Management Guide for Oracle WebLogic Portal</i>.</p> |
| bt_create_database.sql | <p>Creates the WEBLOGIC_EVENT database and WEBLOGIC_EVENT database owner user login, and grants the privileges to the WEBLOGIC_EVENT user necessary for creating database objects.</p> <p>Following are the default names:</p> <ul style="list-style-type: none"> <li>■ Database name: WEBLOGIC_EVENT</li> <li>■ Database owner user: WEBLOGIC_EVENT</li> <li>■ Password: WEBLOGIC_EVENT</li> </ul>  |

---

---

## Using Oracle

This chapter describes the steps necessary to use an Oracle database with WebLogic Portal, and includes the following sections:

- [Section 7.1, "Configuring Oracle Databases"](#)
- [Section 7.2, "Configuring the Main WebLogic Portal Database"](#)
- [Section 7.3, "Manually Creating a Separate Database and Database Objects for Behavior Tracking"](#)
- [Section 7.4, "Database Scripts and Usage Notes"](#)

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

---

---

**Note:** A WebLogic Portal Oracle database user that has DBA privileges causes database create errors and upgrade issues. Before you create database objects or perform an upgrade, you should revoke the Oracle user's DBA privileges and replace them with the following privileges: GRANT CREATE TABLE, CREATE VIEW, CREATE TRIGGER, CREATE SEQUENCE, CREATE SESSION, UNLIMITED TABLESPACE TO <<WEBLOGIC>>.

---

---

### 7.1 Configuring Oracle Databases

Before proceeding, read [Section 2.1, "Overview of Enterprise-Quality Database Configuration for WebLogic Portal."](#)

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

Multiple database users 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). Any additional content management repositories require a separate database user.

---

---

**Note:** Oracle configuration settings can impact Content Search performance; for more information, see [Section 3.3.5, "Improving Content Search Response Time for Oracle."](#)

---

---

To configure an Oracle database:

1. Be sure to back up your database before installing any new database objects. See your database documentation for details.
2. Review the provided sample scripts, located in the <WLPORTAL\_HOME>/portal/db/oracle/admin directory. See [Table 7-1](#) and the comments in the scripts for additional information.
3. Copy and modify the sample scripts appropriately for your environment to create each of the following database schemas:
  - a. Follow the steps in [Section 7.2, "Configuring the Main WebLogic Portal Database"](#) to create the main WebLogic Portal user and database objects.
  - b. If you want to create a separate behavior tracking database, follow the steps in [Section 7.3, "Manually Creating a Separate Database and Database Objects for Behavior Tracking."](#)

## 7.2 Configuring the Main WebLogic Portal Database

To configure the main WebLogic Portal database, follow these steps:

1. If you want to create new tablespaces for the main WebLogic Portal schema:
  - a. Copy the `create_tablespaces.sql` script and modify it appropriately for your environment. See [Table 7-1](#) and the comments in the script for additional information.
  - b. Run the modified `create_tablespaces.sql` script. For example, from SQL\*Plus:

```
@create_tablespaces.sql
```

2. Create the main WebLogic Portal database user:
  - a. Copy the `create_users.sql` script and modify it appropriately for your environment. See [Table 7-1](#) and the comments in the script for additional information.
  - b. Run the modified `create_users.sql` script. For example, from SQL\*Plus:

```
@create_users.sql
```

Follow the remaining steps only if you want to create database objects manually rather than using the Configuration Wizard. To perform the remaining steps using the Configuration Wizard, see *Oracle Fusion Middleware Creating Domains Using the Configuration Wizard*.

1. Open your domain's `database.properties` file for edit.
2. Set `database=oracle`.
3. Update the following settings (by replacing the @ symbols and the text between the symbols with the correct values) for your main WebLogic Portal database:

```
oracle.user=@DB_USER@
oracle.password=@DB_PASSWORD@
oracle.url=jdbc:bea:oracle://@DB_HOST@:@DB_PORT@;SID=@DB_NAME@
```

4. Create the database objects.
  - a. Navigate to the <MW\_HOME>\user\_projects\domains\myPortalDomain directory.
  - b. Enter one of the commands:



- On Windows, double-click `create_db.cmd`.
- On UNIX, run `create_db.sh`.

If any error messages are displayed, check the `create_db.log` file for additional information.

---

**Note:** After running `create_db` script, you need to update the database administrator password. See [Section 3.8, "Note About Creating or Refreshing Database Objects"](#) for detailed information.

---

5. Replace the JDBC data sources in your domain, which point to PointBase by default, with data sources that point to Oracle. You can configure them using the WebLogic Server Administration Console or choose from the samples provided and update them for your database environment. Sample `jdbc.xml` definition files for each database and driver that Oracle supports are available in the `<WLPORTAL_HOME>\portal\db\jdbc\database_driver` directory; for example, `oracle_bea` or `oracle_thin`. Follow the instructions in the `<WLPORTAL_HOME>\portal\db\jdbc\README.txt` file.
6. For improved performance, move indexes to the `WEBLOGIC_INDEX` tablespace by executing `rebuild_indexes.sql`.

---

**Note:** Do this while WebLogic Server is stopped.

---

## 7.3 Manually Creating a Separate Database and Database Objects for Behavior Tracking

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 "Setting Up Events and Behavior Tracking" in the *Oracle Fusion Middleware Interaction Management Guide for Oracle WebLogic Portal*.

---

**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.

---

To create a separate database for behavior tracking:

1. Modify the `bt_create_tablespace.sql` file and the `bt_create_users.sql` file for your environment. See [Table 7-1](#) and the comments in the script for additional information.
2. Run the modified `bt_create_tablespace.sql` script.
3. Run the modified `bt_create_users.sql` script.
4. Navigate to the appropriate database directory based on your environment:  
`<WLPORTAL_HOME>\portal\db\oracle`
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_triggers.sql`
6. Run the following scripts from the path `<WLPORTAL_HOME>\portal\db\data\required`:
    - `bt_insert_system_data.sql`
    - `bt9_insert_system_data.sql`
  7. Use the WebLogic Server Administration Console to configure a non-XA JDBC data source to access your behavior tracking database. Associate the JNDI name `p13n.trackingDataSource` with that data source and then remove `p13n.trackingDataSource` from `p13nDataSource`.

## 7.4 Database Scripts and Usage Notes

Table 7–1 describes the scripts that enable you to configure the Oracle database and create WebLogic Portal objects in that database.

**Table 7–1 Database Scripts and Usage Notes**

| Script Name                      | Description  |
|----------------------------------|--|
| <code>create_users.sql</code>    | <p>Creates a user and password, and sets default and temporary tablespaces. Grants privileges to that user.</p> <p><b>Note:</b> Make a copy of this script and edit it to replace <code>&lt;&lt;WEBLOGIC&gt;&gt;</code> with the user name, password and tablespace names.</p> <p>Typical names for the main WebLogic Portal database are:</p> <ul style="list-style-type: none"> <li>■ Database schema user = <code>WEBLOGIC</code></li> <li>■ Database password = <code>WEBLOGIC</code></li> <li>■ Default tablespace = <code>WEBLOGIC_DATA</code></li> <li>■ Temporary tablespace = <code>TEMP</code></li> </ul> <p><b>Note:</b> If you decide to create an additional content management repository, you must create another schema for it. For additional information, see the <i>Oracle Fusion Middleware Content Management Guide for Oracle WebLogic Portal</i>.</p> |
| <code>rebuild_indexes.sql</code> | Rebuilds schema user ( <code>WEBLOGIC</code> ) indexes to move them from the table ( <code>WEBLOGIC_DATA</code> ) tablespace to the ( <code>WEBLOGIC_INDEX</code> ) tablespace.  |
| <code>statistics.sql</code>      | Runs <code>dbms_utility.analyze_schema</code> to compute database statistics needed for the Oracle optimizer. Update statistics periodically and whenever any significant changes in database data occur.  |
| <code>install_report.sql</code>  | Builds an informational installation report about the database objects created in the schema.  |

**Table 7–1 (Cont.) Database Scripts and Usage Notes**

| Script Name               | Description  |
|---------------------------|--|
| dbsize.sql                | Builds a report showing free space in database tablespaces.  |
| bt_create_tablespaces.sql | Creates the tablespace for behavior event tracking.<br><b>Note:</b> You must edit this script to modify the path names for the <code>EVT_DATA_PATHNAME</code> and <code>DATA_FILENAME</code> variables to reflect valid disk locations.  |
| bt_create_users.sql       | Creates a behavior event tracking schema user and password, and sets default and temporary tablespaces. Grants privileges to that user.<br><b>Note:</b> You can edit this script to change the schema user name, password and tablespace names.<br>The following defaults are used: <ul style="list-style-type: none"> <li>▪ Database user: <code>WEBLOGIC_EVENT</code></li> <li>▪ Password: <code>WEBLOGIC_EVENT</code></li> <li>▪ Default tablespace = <code>WEBLOGIC_EVENT_DATA</code></li> <li>▪ Temporary tablespace = <code>TEMP</code></li> </ul> |

## 7.5 Using WebLogic Portal with Oracle RAC

This section describes the steps necessary to use Oracle Real Application Cluster (RAC) 10g with WebLogic Portal.

Oracle Real Application Cluster is Oracle's database cluster technology. RAC offers scalability, high-availability and load-balancing at the database tier of a WebLogic Portal deployment. For more information on Oracle RAC please refer to the section "Using WebLogic Server with Oracle RAC" in *Oracle Fusion Middleware Configuring and Managing JDBC for Oracle WebLogic Server*.

The following sections address the various requirements and configuration choices when using Oracle RAC with WebLogic Portal. These include:

- [Section 7.5.1, "Supported Configuration"](#)
- [Section 7.5.2, "Configuration Considerations for Oracle RAC"](#)
- [Section 7.5.3, "Procedure for Using Oracle RAC with WebLogic Portal"](#)
- [Section 7.5.4, "Sample Configuration Files"](#)

### 7.5.1 Supported Configuration

For supported Oracle RAC and JDBC driver combinations, and for all the database configurations supported by WebLogic Portal 10.3.2, see "Oracle Fusion Middleware Supported System Configurations."

### 7.5.2 Configuration Considerations for Oracle RAC

For each JDBC data source used by WebLogic Portal, the following configuration choices need to be considered:

- [Section 7.5.2.1, "Multi-Data Source vs. Oracle Thin Driver"](#)
- [Section 7.5.2.2, "Fail-over vs. Load-Balancing"](#)
- [Section 7.5.2.3, "XA Considerations"](#)
- [Section 7.5.2.4, "Supported Configurations"](#)

### 7.5.2.1 Multi-Data Source vs. Oracle Thin Driver

There are two ways to implement JDBC fail-over and load-balancing: through the WebLogic multi-data source, or through the cluster-aware capabilities built into the Oracle Thin Driver. For examples of how these are implemented, see "Using WebLogic Server with Oracle RAC" in *Oracle Fusion Middleware Configuring and Managing JDBC for Oracle WebLogic Server*.

As pointed out in *Using WebLogic Server with Oracle RAC*, there are some limitations if you are using the Oracle Thin Driver alone to access Oracle RAC. Specifically, you cannot use load-balancing together with XA support.

On the other hand, WebLogic multi-data source fully supports XA with load-balancing. Multi-data source also offers benefits such as faster fail-over and automatic fail-back. These make multi-data source the favorable choice to access RAC. The rest of the document is based on using multi-data source.

### 7.5.2.2 Fail-over vs. Load-Balancing

WebLogic Portal data sources support both fail-over and load-balancing with the exception of the `portalDataSourceNeverXA` data source. The `portalDataSourceNeverXA` is used as JDBC Store. As specified in "Using WebLogic Server with Oracle RAC" in *Oracle Fusion Middleware Configuring and Managing JDBC for Oracle WebLogic Server*, the data source of a JDBC Store cannot use the load-balancing algorithm.

### 7.5.2.3 XA Considerations

WebLogic Portal data sources can use either the XA or the non-XA JDBC driver, except `p13nDataSource` and `portalDataSourceNeverXA`. The `p13nDataSource` is used in local JDBC transactions only. The `portalDataSourceNeverXA` is used as JDBC Store. As specified in "Using WebLogic Server with Oracle RAC" in *Oracle Fusion Middleware Configuring and Managing JDBC for Oracle WebLogic Server*, the data source of a JDBC Store cannot use an XA-compliant JDBC driver.

When the multi-data source is used, it is important to make sure that the multi-data source's value of the `global-transaction-protocol` attribute is identical to those of all its underlying physical data sources. For a multi-data source, the value of `global-transaction-protocol` can be configured by either manually editing

```
${DOMAIN_HOME}/config/jdbc/${DATASOURCE_NAME}-jdbc.xml
```

or programmatically through WLST scripts. The WebLogic Administration Console does not provide a way to configure this attribute for a multi-data source.

### 7.5.2.4 Supported Configurations

The supported RAC configurations are summarized in [Table 7-2](#).

**Table 7-2 WebLogic Portal Data Sources When Using Oracle RAC**

| Data Source Name                | Multi-Data Source | MD-Protocol  | XA-Driver | XA- Protocol   |
|---------------------------------|-------------------|--------------|-----------|----------------|
| <code>p13nDataSource</code>     | No                |              | False     | None           |
| <code>cgDataSource</code>       | Yes               | Load-Balance | True      | TwoPhaseCommit |
| <code>cgDataSource-NonXA</code> | No                |              | False     | None           |
| <code>portalDataSource</code>   | Yes               | Load-Balance | True      | TwoPhaseCommit |

**Table 7–2 (Cont.) WebLogic Portal Data Sources When Using Oracle RAC**

| Data Source Name         | Multi-Data Source | MD-Protocol  | XA-Driver | XA- Protocol   |
|--------------------------|-------------------|--------------|-----------|----------------|
| portalDataSourceAlwaysXA | Yes               | Load-Balance | True      | TwoPhaseCommit |
| portalDataSourceNeverXA  | Yes               | Failover     | False     | OnePhaseCommit |
| sampleDataSource         | Yes               | Load-Balance | True      | TwoPhaseCommit |

---

**Note:** The values in the above table are consistent with the JDBC data source configuration files packaged with WebLogic Portal, with the addition of the **Multi-Data Source** column.

---

### 7.5.3 Procedure for Using Oracle RAC with WebLogic Portal

The configuration steps for Oracle RAC are very similar to those described in the previous sections [Section 7.2, "Configuring the Main WebLogic Portal Database"](#) and [Section 7.3, "Manually Creating a Separate Database and Database Objects for Behavior Tracking."](#) Each of these three sections includes seven steps.

1. If you want to create new tablespaces for the main WebLogic Portal schema:

---

**Note:** To run the SQL script in this step, log on to any one node of the RAC. The specific node is not important since all Oracle instances in the RAC share the same disk storage.

---

- a. Copy the `create_tablespaces.sql` script and modify it appropriately for your environment. See [Table 7–1](#) and the comments in the script for additional information.
- b. Run the modified `create_tablespaces.sql` script. For example, from SQL\*Plus:

```
@create_tablespaces.sql
```

2. Create the main WebLogic Portal database user:

---

**Note:** To run the SQL script in this step, log on to any one node of the RAC. The specific node is not important since all Oracle instances in the RAC share the same disk storage.

---

- a. Copy the `create_users.sql` script and modify it appropriately for your environment. See [Table 7–1](#) and the comments in the script for additional information.
- b. Run the modified `create_users.sql` script. For example, from SQL\*Plus:

```
@create_users.sql
```

Follow the remaining steps only if you want to create database objects manually rather than using the Configuration Wizard. To perform the remaining steps using the Configuration Wizard, see *Oracle Fusion Middleware Creating Domains Using the Configuration Wizard*.

3. Open your domain's `database.properties` file for edit.

4. Set `database=oracle`.
5. Update the following settings (by replacing the @ symbols and the text between the symbols with the correct values) for your main WebLogic Portal database:

```
oracle.user=@DB_USER@
oracle.password=@DB_PASSWORD@
oracle.url=jdbc:bea:oracle://@DB_HOST@:@DB_PORT@;SID=@DB_NAME@
```

---

---

**Note:** In the `oracle.url` string, replace `@DB_HOST@` and `@DB_NAME@` with the hostname and SID of any one node in the RAC. Since all Oracle instances in the RAC share the same disk storage, the database objects created in step 6 will be visible from all other nodes in the RAC.

---

---

6. Create the database objects.
  - a. Navigate to the `<MW_HOME>\user_projects\domains\myPortalDomain` directory.
  - b. Enter one of the commands:
    - On Windows, double-click `create_db.cmd`.
    - On UNIX, run `create_db.sh`.

If any error messages are displayed, check the `create_db.log` file for additional information.

---

---

**Note:** After running `create_db` script, you need to update the database administrator password. See [Section 3.8, "Note About Creating or Refreshing Database Objects"](#) for detailed information.

---

---

7. Replace the JDBC data sources in your domain with multi-data sources pointing to your Oracle RAC.
  - a. For each JDBC data source, create one physical data source pointing to each node in the RAC. For example, for `p13nDataSource`, create physical data sources `p13nDataSource-1` to `p13nDataSource-N`, each pointing to one RAC node. Use the WebLogic Server Administration Console to create these data sources.

Alternatively, you could create them by copying and updating the sample `jdbc.xml` files in the `<WLPORTAL_HOME>/portal/db/jdbc/oracle_thin` directory following the steps below:

- 1) Copy the `dataSourceName-jdbc.xml` file to `<DOMAIN_HOME>/config/jdbc` directory, and name it `dataSourceName-n-jdbc.xml`, where `n` is the sequence number of the `n`-th RAC node.
- 2) Edit the file `dataSourceName-n-jdbc.xml`. Replace `@DB_USER@`, `@DB_USER_PASSWD@`, `@DB_HOST@`, `@DB_PORT@` and `@DB_NAME@`. For `@DB_HOST@` and `@DB_NAME@`, use the hostname and SID of the `n`-th RAC node.
- 3) Edit `config.xml` to add this newly created JDBC data source as a JDBC system resource.

- b. Create a multi-data source for each of the JDBC data sources, which encompasses the physical data sources created above. For example, create a `p13nDataSource` multi-data source which includes `p13nDataSource-1` to `p13nDataSource-N` created above.

If you use the WebLogic Server Administration Console, you would need to first remove the existing data source with the same name, and then create the multi-data source. Also, since the WebLogic Server Administration Console does not provide a way to configure the value of `global-transaction-protocol` attribute for a multi-data source, you need to manually add the attribute into the corresponding `jdbc.xml` later.

Alternatively, you could manually edit the corresponding `jdbc.xml` following the steps below:

1) Open the corresponding `jdbc.xml` file, delete the `jdbc-driver-params` element and the `jdbc-connection-pool-params` element, including all their children elements:

2) Between the `jndi-name` and the `global-transaction-protocol` elements, add the following two elements. Determine the value of `algorithm-type` from [Example 7-1](#). The value of `data-source-list` is a comma-separated list of all the underlying physical data sources.

```
<algorithm-type>Load-Balancing</algorithm-type>
```

or

```
<algorithm-type>Failover</algorithm-type>
<data-source-list>portalDataSourceAlwaysXA-1,
portalDataSourceAlwaysXA-2 </data-source-list>
```

- c. An alternative to the manual procedure described above in steps 1 and 2 is WLST scripting. The sample `build.xml`, `oracconf.py` and `oracconf.py.properties` provide an example of how to configure an existing WebLogic Portal domain to point to an Oracle RAC database. These files are listed in the WebLogic Portal 9.2 document Appendix B, Sample WLST Scripts at [http://download.oracle.com/docs/cd/E13218\\_01/wlp/docs92/db/appx\\_oracle\\_rac\\_scripts.html](http://download.oracle.com/docs/cd/E13218_01/wlp/docs92/db/appx_oracle_rac_scripts.html). To use the sample, follow the steps below:

1) Download these files into your `DOMAIN_HOME` directory.

2) Open the file `oracconf.py.properties`. Edit the values in the "Database usernames and passwords" and the "ORACLE RAC configuration" sections to reflect your Oracle RAC environment.

3) Open a command prompt, change current directory to `DOMAIN_HOME/bin`. Execute the following script:

```
./setDomainEnv.sh (or setDomainEnv.cmd)
```

4) Execute `ant`.

## 7.5.4 Sample Configuration Files

### Example 7-1 p13nDataSource Multi-Data Source

```
<?xml version="1.0" encoding="UTF-8"?>
<jdbc-data-source xmlns="http://www.bea.com/ns/weblogic/90">
  <name>p13nDataSource</name>
```

```

<jdbc-data-source-params>
  <jndi-name>p13n.trackingDataSource</jndi-name>
  <jndi-name>p13n.sequencerDataSource</jndi-name>
  <jndi-name>cm.sequencerDataSource</jndi-name>
  <jndi-name>p13n.leasemanager</jndi-name>
  <jndi-name>p13n.dataSyncDataSource</jndi-name>
  <jndi-name>p13n.entitlementsDataSource</jndi-name>
  <jndi-name>p13n.quiescenceDataSource</jndi-name>
  <algorithm-type>Load-Balancing</algorithm-type>
  <data-source-list>p13nDataSource-1,p13nDataSource-2</data-source-list>
  <global-transactions-protocol>None</global-transactions-protocol>
</jdbc-data-source-params>
</jdbc-data-source>

```

### Example 7-2 p13nDataSource-1 Data Source

```

<?xml version="1.0" encoding="UTF-8"?>
<jdbc-data-source xmlns="http://www.bea.com/ns/weblogic/90">
  <name>p13nDataSource-1</name>
  <jdbc-driver-params>
    <url>jdbc:oracle:thin:@rnhp380-c11-23-vip:1521:DBSRAC101</url>
    <driver-name>oracle.jdbc.OracleDriver</driver-name>
    <properties>
      <property>
        <name>user</name>
        <value>WEBLOGIC_8</value>
      </property>
    </properties>
    <password-encrypted>{3DES}cIUM0gs5Divb+Uw1IFgSoA==</password-encrypted>
  </jdbc-driver-params>
  <jdbc-connection-pool-params>
    <initial-capacity>5</initial-capacity>
    <max-capacity>20</max-capacity>
    <capacity-increment>1</capacity-increment>
    <test-connections-on-reserve>true</test-connections-on-reserve>
    <test-table-name>SQL SELECT 1 FROM DUAL</test-table-name>
  </jdbc-connection-pool-params>
  <jdbc-data-source-params>
    <jndi-name>p13nDataSource-1</jndi-name>
    <global-transactions-protocol>None</global-transactions-protocol>
  </jdbc-data-source-params>
</jdbc-data-source>

```

### Example 7-3 p13nDataSource-2 Data Source

```

<?xml version="1.0" encoding="UTF-8"?>
<jdbc-data-source xmlns="http://www.bea.com/ns/weblogic/90">
  <name>p13nDataSource-2</name>
  <jdbc-driver-params>
    <url>jdbc:oracle:thin:@rnhp380-c11-25-vip:1521:DBSRAC102</url>
    <driver-name>oracle.jdbc.OracleDriver</driver-name>
    <properties>
      <property>
        <name>user</name>
        <value>WEBLOGIC_8</value>
      </property>
    </properties>
    <password-encrypted>{3DES}cIUM0gs5Divb+Uw1IFgSoA==</password-encrypted>
  </jdbc-driver-params>

```



```
<jdbc-connection-pool-params>
  <initial-capacity>5</initial-capacity>
  <max-capacity>20</max-capacity>
  <capacity-increment>1</capacity-increment>
  <test-connections-on-reserve>true</test-connections-on-reserve>
  <test-table-name>SQL SELECT 1 FROM DUAL</test-table-name>
</jdbc-connection-pool-params>
<jdbc-data-source-params>
  <jndi-name>p13nDataSource-2</jndi-name>
  <global-transactions-protocol>None</global-transactions-protocol>
</jdbc-data-source-params>
</jdbc-data-source>
```



---

---

# Using Sybase

This chapter describes the steps necessary to use a Sybase database with WebLogic Portal, and includes the following sections:

- [Section 8.1, "Configuring Sybase Databases"](#)
- [Section 8.2, "Configuring the Main WebLogic Portal Database"](#)
- [Section 8.3, "Manually Creating a Separate Database and Database Objects for Behavior Tracking"](#)
- [Section 8.4, "Database Scripts and Usage Notes"](#)

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

## 8.1 Configuring Sybase Databases

Before proceeding, read [Section 2.1, "Overview of Enterprise-Quality Database Configuration for WebLogic Portal."](#)

The database creation scripts install domain-specific tables. It is recommended that you work with your database administrator to modify the sample scripts, and to create the database users and devices needed for your Sybase environment.

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 enterprise-quality system from a single Sybase installation). Any additional content management repositories require a separate user.

---



---

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

If a Sybase instance is defined with a page size smaller than 8K, the WebLogic Portal tables are created but warning messages may be issued at creation time, indicating that the row size could exceed the row size limit. These warnings may result in runtime 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.

---



---

To configure a Sybase database:

1. Be sure to back up your database before installing any new database objects. See your database vendor documentation for details.
2. Review the provided sample scripts, located in the <WLPORTAL\_HOME>\portal\db\sybase\admin directory. See [Table 8-1](#) and the comments in the scripts for additional information.
3. Copy and modify the sample scripts appropriately for your environment to create each of the following databases:
  - a. Follow the steps in [Section 8.2, "Configuring the Main WebLogic Portal Database"](#) to create the main WebLogic Portal database and database objects.
  - b. If you want to create a separate behavior tracking database, follow the steps in [Section 8.3, "Manually Creating a Separate Database and Database Objects for Behavior Tracking."](#)

## 8.2 Configuring the Main WebLogic Portal Database

To configure the main WebLogic Portal database, follow these steps:

1. Copy the `create_devices.sql` script and modify it appropriately for your environment. See [Table 8-1](#) and the comments in the script for additional information.
2. Run the modified `create_devices.sql` as a user with system administrator privileges (normally the `sa` user). For example, using `isql`:
 

```
isql -Usa -SMYSYBASE -e -icreate_devices.sql -ocreate_devices.log
```
3. Copy the `create_database.sql` script and modify it appropriately for your environment. See [Table 8-1](#) and the comments in the script for additional information.
4. Run the modified `create_database.sql` as a user with system administrator privileges (normally the `sa` user). For example, using `isql`:
 

```
isql -Usa -SMYSYBASE -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.

Follow the remaining steps only if you want to create database objects manually rather than using the Configuration Wizard. To perform the remaining steps using the Configuration Wizard, see *Oracle Fusion Middleware Creating Domains Using the Configuration Wizard*.

1. Open your domain's `database.properties` file for edit.
2. Set `database=sybase`.
3. Update the following settings (by replacing the @ symbols and the text between the symbols with the correct values) for your main WebLogic Portal database:

```
sybase.user=@DB_USER@
sybase.password=@DB_PASSWORD@
sybase.url=jdbc:bea:sybase://@DB_HOST@:@DB_PORT@;DatabaseName=@DB_NAME@
```

4. Create the database objects. You can use the Configuration Wizard or follow these steps:
  - a. Navigate to the `<MW_HOME>\user_projects\domains\myPortalDomain` directory.
  - b. Enter one of the commands:
    - On Windows, double-click `create_db.cmd`.
    - On UNIX, run `create_db.sh`.

If any error messages are displayed, check the `create_db.log` file for additional information.

---

**Note:** After running `create_db` script, you need to update the database administrator password. See [Section 3.8, "Note About Creating or Refreshing Database Objects"](#) for detailed information.

---

5. Replace the JDBC data sources in your domain, which point to PointBase by default, with data sources that point to SQL Server. You can configure them using the WebLogic Server Administration Console or choose from the samples provided and update them for your database environment. Sample `jdbc.xml` definition files for each database and driver that Oracle supports are available in the `<WLPORTAL_HOME>\portal\db\jdbc\database_driver` directory; for example, `sybase_bea`. Follow the instructions in the `<WLPORTAL_HOME>\portal\db\jdbc\README.txt` file.

## 8.3 Manually Creating a Separate Database and Database Objects for Behavior Tracking

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 "Setting Up Events and Behavior Tracking" in the *Oracle Fusion Middleware Interaction Management Guide for Oracle WebLogic Portal*.

---

---

**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.

---

---

To create a separate database for behavior tracking:

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 scripts and in [Table 8-1](#).
2. Run `bt_create_devices.sql` as a user with system administrator privileges. For example, using `isql`:  

```
isql -Usa -SMYSYBASE -e -ibt_create_devices.sql -obt_create_devices.log
```
3. Run `bt_create_database.sql` as a user with system administrator privileges. For example, using `isql`:  

```
isql -Usa -SMYSYBASE -e -ibt_create_database.sql -obt_create_database.log
```
4. Navigate to the appropriate database directory based on your environment; for example, `<WLPORTAL_HOME>\portal\db\sybase`.
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_triggers.sql`
6. Run the following scripts from the path `<WLPORTAL_HOME>\portal\db\data\required`:
  - `bt_insert_system_data.sql`
  - `bt9_insert_system_data.sql`
7. Use the WebLogic Server Administration Console to configure a non-XA JDBC data source to access your behavior tracking database. Associate the JNDI name `p13n.trackingDataSource` with that data source and then remove `p13n.trackingDataSource` from `p13nDataSource`.

## 8.4 Database Scripts and Usage Notes

[Table 8-1](#) describes the scripts that enable you to configure the Sybase database.

**Table 8–1 Database Scripts and Usage Notes**

| Script Name        | Description   |
|--------------------|---|
| create_devices.sql | <p data-bbox="764 275 1446 348">Creates database devices. Database devices must be created by a user with system administrator privileges (normally the sa user).</p> <ol data-bbox="764 369 1446 800" style="list-style-type: none"> <li data-bbox="764 369 1446 443">1. Make a copy of this script and edit it to replace &lt;&lt;WEBLOGIC&gt;&gt; with the appropriate database name and device names for each database you create.</li> <li data-bbox="764 464 1446 537">2. Edit the script to reflect valid disk locations for D: \DATAFILE, E: \LOGFILE, and F: \INDEXFILE device locations.</li> <li data-bbox="764 558 1446 800">3. Modify file sizes as needed depending on the WebLogic Portal functionality you are using. The following values for a production system are provided as a general guide, but you can adjust these values as needed (for example, an individual developer's database can have smaller files):<br/> log segment: 2048mb<br/> data segment: 2048mb<br/> index segment: 500mb</li> </ol> <p data-bbox="764 821 1446 894">Optimally, locate data, log, and index devices on separate physical disks (with separate controllers) and away from any system database files, unless you are using RAID devices.</p> <p data-bbox="764 915 1446 968"><b>Note:</b> Do not change the name of the WEBLOGIC_INDEX index segment.</p> <p data-bbox="764 989 1446 1041">Typical device names for the main WebLogic Portal database are:</p> <ul data-bbox="764 1062 1446 1106" style="list-style-type: none"> <li data-bbox="764 1062 1446 1083">■ Data device: WEBLOGIC_DATA</li> <li data-bbox="764 1104 1446 1125">■ Log device: WEBLOGIC_LOG</li> </ul> |

**Table 8–1 (Cont.) Database Scripts and Usage Notes**

| Script Name   | Description  |
|---|--|
| create_database.sql                                       | <p>Creates the database and login. You must create a main WebLogic Portal database.</p> <p>You must also create a database for each additional content management repository.</p> <p>The devices created by <code>create_devices.sql</code> are used and indexes are placed in their own <code>WEBLOGIC_INDEX</code> segment. An alias is added to the database owner (<code>dbo</code>) user of the database.</p> <p>Make a copy of this script and edit it to replace <code>&lt;&lt;WEBLOGIC&gt;&gt;</code> with the appropriate database name, database user, and password for each database you create. Also adjust database and log sizes as appropriate.</p> <p>If the database you are creating is a development database (and therefore database recovery is not a concern), you can uncomment and set the <code>truncate log on checkpoint</code> option to <code>true</code>.</p> <p>If your applications will use Oracle Enterprise Pack for Eclipse (OEPE) for create database or RowSet controls, uncomment and set the <code>DDL in transaction</code> option to <code>true</code>; otherwise create commands will not work properly.</p> <p>Typical names for the main WebLogic Portal database are:</p> <ul style="list-style-type: none"> <li>■ Database name: <code>WEBLOGIC</code></li> <li>■ Database schema user: <code>WEBLOGIC</code></li> <li>■ Password: <code>WEBLOGIC</code></li> </ul> <p><b>Note:</b> Do not change the name of the <code>WEBLOGIC_INDEX</code> index segment.</p> <p>When you run the script with these values, it creates the <code>WEBLOGIC</code> database, the <code>WEBLOGIC_INDEX</code> file group, and <code>WEBLOGIC</code> database owner (<code>dbo</code>) user login. An alias is created to make <code>WEBLOGIC</code> the <code>dbo</code> user in the database. It also sets the <code>WEBLOGIC</code> database as the default database for the <code>WEBLOGIC</code> user.</p> <p><b>Note:</b> If you decide to create an additional content management repository, you must create a database with different user names for it. See the <i>Oracle Fusion Middleware Content Management Guide for Oracle WebLogic Portal</i>.</p> |
| statistics_build.sql                                      | <p>Builds <code>statistics.sql</code> to update table and index statistics for the database optimizer. Update statistics periodically and whenever any significant changes in database data occur.</p>   |
| install_report_build.sql<br>install_report_<br>static.sql | <p>Builds an informational installation report about the database objects created by the <code>WEBLOGIC</code> user.</p>   |



**Table 8–1 (Cont.) Database Scripts and Usage Notes**

| Script Name           | Description   |
|-----------------------|---|
| bt_create_devices.sql | <p data-bbox="764 260 1430 338">Creates behavior tracking database devices. Database devices must be created by a user with system administrator privileges (normally the <i>sa</i> user).</p> <p data-bbox="764 352 1430 512">You must also edit the script to reflect valid disk locations for D:\DATAFILE, E:\LOGFILE, and F:\INDEXFILE device locations; you may also need to modify file sizes. Optimally, locate data, log, and index devices on separate physical disks (with separate controllers) and away from any system database files, unless you are using RAID devices.</p> <p data-bbox="764 527 1166 554">The following default names are used:</p> <ul data-bbox="764 569 1219 638" style="list-style-type: none"><li data-bbox="764 569 1219 596">■ Data device: WEBLOGIC_EVENT_DATA</li><li data-bbox="764 604 1219 638">■ Log device: WEBLOGIC_EVENT_LOG</li></ul> |



This chapter describes the steps necessary to use a DB2 database with WebLogic Portal, and includes the following sections:

- [Section 9.1, "Configuring DB2 Databases"](#)
- [Section 9.2, "Configuring the Main WebLogic Portal Database"](#)
- [Section 9.3, "Manually Creating a Separate Database and Database Objects for Behavior Tracking"](#)
- [Section 9.4, "Database Scripts and Usage Notes"](#)

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

## 9.1 Configuring DB2 Databases

Before proceeding, read [Section 2.1, "Overview of Enterprise-Quality Database Configuration for WebLogic Portal."](#)

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

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). Any additional content management repositories require a separate user.

---

---

**Note:** To ensure that portal applications run correctly on DB2, you must set some minimum configuration parameters. If you do not, heavy portal activity might exceed database capacity.

---

---

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

To configure a DB2 database:

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

2. Review the provided sample scripts, located in the <WLPORTAL\_HOME>/portal/db/oracle/admin directory. See [Table 9–1](#) and the comments in the scripts for additional information.
3. Copy and modify the sample scripts appropriately for your environment to create each of the following database schemas:
  - a. Follow the steps in [Section 9.2, "Configuring the Main WebLogic Portal Database"](#) to create the main WebLogic Portal user and database objects.
  - b. If you want to create a separate behavior tracking database, follow the steps in [Section 9.3, "Manually Creating a Separate Database and Database Objects for Behavior Tracking."](#)

## 9.2 Configuring the Main WebLogic Portal Database

To configure the main WebLogic Portal database, follow these steps:

1. If you do not already have an 8K buffer pool to use:
  - a. Copy the `create_bufferpool.sql` script and modify it appropriately for your environment. See [Table 9–1](#) and the comments in the script for additional information.
  - b. Connect to the database that you want to work with. For example, from the CLP:

```
Db2 connect to database user username password password
```
  - c. Run `create_bufferpool.sql`. For example, from the CLP:

```
Db2 -tf create_bufferpool.sql -v
```
  - d. Restart your database instance.
2. If you do not have an existing an 8K temporary tablespace, or you want to a create new 8K temporary tablespace:
  - a. Copy the `create_temp_tablespace.sql` script and modify it appropriately for your environment. See [Table 9–1](#) and the comments in the script for additional information.
  - b. Run `create_temp_tablespace.sql`. For example, from the CLP:

```
Db2 -tf create_temp_tablespace.sql -v
```
3. If you do not have existing 4K and 8K regular tablespaces, or you want to a create new regular tablespaces:
  - a. Copy the `create_tablespace.sql` script and modify it appropriately for your environment. See [Table 9–1](#) and the comments in the script for additional information.
  - b. Run `create_tablespace.sql`. For example, from the CLP:

```
Db2 -tf create_tablespace.sql -v
```
4. Create the main WebLogic Portal database user:
  - a. Copy the `create_users.sql` script and modify it appropriately for your environment. See [Table 9–1](#) and the comments in the script for additional information.
  - b. Run `create_user.sql`. For example, from the CLP:

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

Follow the remaining steps only if you want to create database objects manually rather than using the Configuration Wizard. To perform the remaining steps using the Configuration Wizard, see *Oracle Fusion Middleware Creating Domains Using the Configuration Wizard*.

5. Open your domain's `database.properties` file for edit.
6. Set `database=db2`.
7. Update the following settings (by replacing the @ symbols and the text between the symbols with the correct values) for your main WebLogic Portal database:

```
db2.user=@DB_USER@
db2.password=@DB_PASSWORD@
db2.url=jdbc:bea:db2://@DB_HOST@:@DB_PORT@;DatabaseName=@DB_NAME@
```

8. Create the database objects.
  - a. Navigate to the `<MW_HOME>\user_projects\domains\myPortalDomain` directory.
  - b. Enter one of the commands:
    - On Windows, double-click `create_db.cmd`.
    - On UNIX, run `create_db.sh`.

If any error messages are displayed, check the `create_db.log` file for additional information.

---

**Note:** After running `create_db` script, you need to update the database administrator password. See [Section 3.8, "Note About Creating or Refreshing Database Objects"](#) for detailed information.

---

9. Replace the JDBC data sources in your domain, which point to PointBase by default, with data sources that point to DB2. You can configure them using the WebLogic Server Administration Console or choose from the samples provided and update them for your database environment. Sample `jdbc.xml` definition files for each database and driver that Oracle supports are available in the `<WLPORTAL_HOME>\portal\db\jdbc\database_driver` directory; for example, `db2_bea`. Follow the instructions in the `<WLPORTAL_HOME>\portal\db\jdbc\README.txt` file.

## 9.3 Manually Creating a Separate Database and Database Objects for Behavior Tracking

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 "Setting Up Events and Behavior Tracking" in the *Oracle Fusion Middleware Interaction Management Guide for Oracle WebLogic Portal*.

---

**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.

---

To create a separate database for behavior tracking:

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 scripts and in [Table 9-1](#).
2. From the CLP, run the `bt_create_tablespaces.sql` script. For example:  

```
Db2 -tf bt_create_tablespaces.sql -v
```
3. From the CLP, run the `bt_create_users.sql` script. For example:  

```
Db2 -tf bt_create_users.sql -v
```
4. Navigate to the appropriate database directory based on your environment:  

```
<WLPORAL_HOME>\portal\db\db2
```
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_triggers.sql`
6. Run the following scripts from the path `<WLPORAL_HOME>\portal\db\data\required`:
  - `bt_insert_system_data.sql`
  - `bt9_insert_system_data.sql`
7. Use the WebLogic Server Administration Console to configure a non-XA JDBC data source to access your behavior tracking database. Associate the JNDI name `p13n.trackingDataSource` with that data source and then remove `p13n.trackingDataSource` from `p13nDataSource`.

## 9.4 Database Scripts and Usage Notes

[Table 9-1](#) describes the scripts that enable you to configure the Oracle database and create WebLogic Portal objects in that database.

**Table 9–1 Database Scripts and Usage Notes**

| Script Name                 | Description  |
|-----------------------------|--|
| create_user.sql             | <p>Grants DB2 <code>createtab</code>, <code>bindadd</code>, and <code>connect</code> privileges to the schema owner user.</p> <p>Because IBM DB2 databases authenticate users using the operating system, you must create an operating system user that owns database schema objects.</p> <p><b>Note:</b> Make a copy of this script and edit it to replace &lt;&lt;WEBLOGIC&gt;&gt; with the schema user name.</p> <p>For the main WebLogic Portal database, the schema user name is typically <code>WEBLOGIC</code>.</p> <p><b>Note:</b> If you decide to create an additional content management repository, you must create a schema with different user names for it. For additional information, see the <i>Oracle Fusion Middleware Content Management Guide for Oracle WebLogic Portal</i>.</p>  |
| create_bufferpool.sql       | <p>Creates an 8K buffer pool, if one does not already exist. You must stop and restart DB2 to utilize new buffer pools.</p> <p>Edit the script to change the 8K buffer pool name. The default buffer pool name is <code>BP8K</code>.</p>   |
| create_tablespaces.sql      | <p>Creates 4K and 8K regular tablespaces. If you want to use existing regular tablespaces of the correct sizes, you do not need to run this script. You can use new or existing tablespaces for the main WebLogic Portal database and any additional content management repositories.</p> <p><b>Note:</b> Make a copy of this script and edit it to replace &lt;&lt;WEBLOGIC&gt;&gt; with the appropriate tablespace and user names. You must also edit the script to reflect valid disk locations for your environment.</p> <p>Typical names for the main WebLogic Portal database are:</p> <ul style="list-style-type: none"> <li>■ Tablespace for tables with a rowsize smaller than 4K: <code>WEBLOGIC_DATA_4K</code></li> <li>■ Tablespace for tables with a rowsize larger than 4K and smaller than 8K: <code>WEBLOGIC_DATA_8K</code></li> <li>■ Database user: <code>WEBLOGIC</code></li> </ul> |
| create_temp_tablespaces.sql | <p>Creates an 8K temporary tablespace. If you want to use an existing temporary tablespace of the correct size, you do not need to run this script. You can use new or existing tablespaces for the main WebLogic Portal database and any additional content management repositories.</p> <p>The default temporary tablespace name is <code>TEMPSPACE_8K</code>.</p> <p><b>Note:</b> Edit this script to use the correct buffer pool name. You must also edit the script to reflect valid disk locations for your environment.</p>   |
| statistics_build.sql        | <p>Builds a file of <code>runstats</code> commands for each table that will compute database statistics needed for the database optimizer. Update statistics periodically and whenever any significant changes in database data occur.</p>   |

**Table 9–1 (Cont.) Database Scripts and Usage Notes**

| <b>Script Name</b>       | <b>Description</b>   |
|--------------------------|--|
| 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.<br><b>Note:</b> Edit the script to specify valid physical disk locations for your environment (event_container), and to use a buffer pool other than the DB2 default buffer pool.  |
| bt_create_users.sql      | Creates the WEBLOGIC_EVENT schema owner user and grants DB2 createtab, bindadd, and connect privileges to the schema owner user.<br><br>Because IBM DB2 databases authenticate users using the operating system, you must create an operating system user that owns database schema objects.<br><br>Edit the script to change the schema user name and system administrator password.<br><br>The default schema owner user name is WEBLOGIC_EVENT. |



---

---

## Data Dictionary

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

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.  
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 WebLogic Portal.

---

---

The New or Changed Version column indicates the version of WebLogic Portal in which a table or column was added or altered. If the New or Changed Version column does not exist, the table or column was created prior to the 9.2 or 10.0 release of WebLogic Portal.

This chapter contains the following sections:

- [Section 10.1, "Personalization Database Objects"](#)
  - [Section 10.1.1.11, "Credential Service Dictionary Tables"](#)
  - [Section 10.1.1.16, "Job Scheduler Dictionary Tables"](#)
  - [Section 10.1.1.22, "Entitlement Reference Dictionary Tables"](#)
- [Section 10.2, "Portal Services Database Objects"](#)

- [Section 10.3, "Portal Framework Database Objects"](#)
- [Section 10.4, "WSRP Portal Framework Database Objects"](#)
- [Section 10.5, "Communities Framework Database Objects"](#)
- [Section 10.6, "Content Management Database Objects"](#)
- [Section 10.7, "Content Management Virtual Content Repository Database Objects"](#)
- [Section 10.8, "RDBMS Security Store Tables"](#)

---

**Note:** The appendix [Appendix A, "Scripts and Properties Files"](#) identifies the filenames and location of DDL (database definition language) files for each set of WebLogic Portal database objects.

---

## 10.1 Personalization Database Objects

Personalization enables you to develop, manage, and measure personalized portal applications. These functions help you target content to a desired audience. This includes the tracking of anonymous users and behavior tracking.

For an entity relation diagram of the Personalization database tables, see [Figure B-1, "Personalization Database Tables"](#).

### 10.1.1 Base Personalization Database Tables

The following tables support basic personalization functionality:

- [Section 10.1.1.1, "USER\\_PROFILE Database Table"](#)
- [Section 10.1.1.2, "ENTITY Database Table"](#)
- [Section 10.1.1.3, "PROPERTY\\_KEY Database Table"](#)
- [Section 10.1.1.4, "PROPERTY\\_VALUE Database Table"](#)
- [Section 10.1.1.5, "SEQUENCER Database Table"](#)
- [Section 10.1.1.6, "USERS Database Table"](#)
- [Section 10.1.1.7, "GROUPS Database Table"](#)
- [Section 10.1.1.8, "GROUPMEMBERS Database Table"](#)
- [Section 10.1.1.9, "P13N\\_QUIESCENT\\_STATE Database Table"](#)
- [Section 10.1.1.10, "P13N\\_LEASE Database Table"](#)
- [Section 10.1.2.1, "P13N\\_ANONYMOUS\\_PROPERTY Database Table"](#)
- [Section 10.1.2.2, "P13N\\_ANONYMOUS\\_USER Database Table"](#)
- [Section 10.1.1.11, "Credential Service Dictionary Tables"](#)
  - [Section 10.1.1.12, "P13N\\_CREDENTIAL Database Table"](#)
  - [Section 10.1.1.13, "P13N\\_CREDENTIAL\\_APPLICATION Database Table"](#)
  - [Section 10.1.1.14, "P13N\\_CREDENTIAL\\_MASTER Database Table"](#)
  - [Section 10.1.1.15, "P13N\\_CREDENTIAL\\_METADATA Database Table"](#)
- [Section 10.1.1.16, "Job Scheduler Dictionary Tables"](#)
  - [Section 10.1.1.17, "P13N\\_JOB\\_DEFINITION Database Table"](#)

- Section 10.1.1.18, "P13N\_JOB\_LISTENER Database Table"
- Section 10.1.1.19, "P13N\_JOB\_PARAMETER Database Table"
- Section 10.1.1.20, "P13N\_JOB\_RUN\_HEADER Database Table"
- Section 10.1.1.21, "P13N\_JOB\_RUN\_DATA Database Table"
- Section 10.1.1.22, "Entitlement Reference Dictionary Tables"
  - Section 10.1.1.23, "P13N\_ENTITLEMENT\_APPLICATION Database Table"
  - Section 10.1.1.24, "P13N\_ENTITLEMENT\_POLICY Database Table"
  - Section 10.1.1.25, "P13N\_ENTITLEMENT\_RESOURCE Database Table"
  - Section 10.1.1.26, "P13N\_ENTITLEMENT\_ROLE Database Table"
  - Section 10.1.1.27, "P13N\_DELEGATED\_HIERARCHY Database Table"
- Section 10.1.3, "Behavior Tracking Database Tables"
- Section 10.1.4, "Data Synchronization Database Tables"

### 10.1.1.1 USER\_PROFILE Database Table

This table associates users with profiles. User profiles use property sets to organize the properties that they contain.

**Table 10–1** *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 |
| CREATION_DATE | DATE         | Not Null   | The date and time this record was created. |

### 10.1.1.2 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 10–2** *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.                        |

### 10.1.1.3 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 10–3** *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.                            |
| 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.  |

#### 10.1.1.4 PROPERTY\_VALUE Database Table

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

**Table 10–4** *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:<br>0=Boolean, 1=Integer, 2=Float, 3=Text,<br>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.   |
| 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.  |

#### 10.1.1.5 SEQUENCER Database Table

The SEQUENCER table maintains all of the sequence identifiers (for example, property\_meta\_data\_id\_sequence, and so on) used in the application.

**Table 10–5 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, and has a value of 0 or 1. This column is used as a generic locking mechanism that can be used for multiple database environments. |

#### 10.1.1.6 USERS Database Table

This table was added in WebLogic Portal version 9.2. It maintains user, group, and password information for the WebLogic Server SQLAuthenticator.

**Table 10–6 USERS Table Metadata**

| Column Name   | Data Type     | Null Value | Description                   | New or Changed Version |
|---------------|---------------|------------|-------------------------------|------------------------|
| U_NAME        | VARCHAR(200)  | Not Null   | PK – Unique name of the user. | 9.2                    |
| U_PASSWORD    | VARCHAR(50)   | Not Null   | Password of the user.         | 9.2                    |
| U_DESCRIPTION | VARCHAR(1000) | Null       | Optional description.         | 9.2                    |

#### 10.1.1.7 GROUPS Database Table

This table was added in WebLogic Portal version 9.2. It maintains information on groups and their descriptions.

**Table 10–7 GROUPS Table Metadata**

| Column Name   | Data Type     | Null Value | Description                    | New or Changed Version |
|---------------|---------------|------------|--------------------------------|------------------------|
| G_NAME        | VARCHAR(200)  | Not Null   | PK – Unique name of the group. | 9.2                    |
| G_DESCRIPTION | VARCHAR(1000) | Null       | Optional description.          | 9.2                    |

#### 10.1.1.8 GROUPMEMBERS Database Table

This table was added in WebLogic Portal version 9.2. It maintains group hierarchy information.

**Table 10–8 GROUPMEMBERS Table Metadata**

| Column Name | Data Type    | Null Value | Description  | New or Changed Version |
|-------------|--------------|------------|--|------------------------|
| G_NAME      | VARCHAR(200) | Not Null   | PK and FK to the GROUPS table. Unique name of the group. | 9.2                    |
| G_MEMBER    | VARCHAR(200) | Not Null   | PK – Member of the group.                                | 9.2                    |

### 10.1.1.9 P13N\_QUIESCENT\_STATE Database Table

This table was added in WebLogic Portal version 9.2. It allows an Administration Console user or visitor tools user to disable modifications to portal resources, and allows you to send multicast messages to a member of a cluster.

**Table 10–9 P13N\_QUIESCENT\_STATE Table Metadata**

| Column Name            | Data Type    | Null Value | Description  | New or Changed Version |
|------------------------|--------------|------------|--|------------------------|
| ENTAPP_NAME            | VARCHAR(255) | Not Null   | PK – Name of the enterprise application.   | 9.2                    |
| WEBAPP_NAME            | VARCHAR(255) | Not Null   | PK – Name of the web application.  | 9.2                    |
| RESOURCE_NAME          | VARCHAR(255) | Not Null   | PK – The name of an entitled or delegated resource; for example, CONTENT, PORTAL.                        | 9.2                    |
| CAPABILITY_NAME        | VARCHAR(80)  | Not Null   | PK – The name of allowed operations/capabilities; anything that allows you to the manipulate the system. | 9.2                    |
| MODE_NAME              | VARCHAR(80)  | Not Null   | The name of the quiescence mode; for example, BLOCKED.   | 9.2                    |
| MODE_ENABLED_USER_NAME | VARCHAR(200) | Not Null   | The name of the person who turned on quiescence for DT, or who set the BLOCKED mode for the resource.    | 9.2                    |

### 10.1.1.10 P13N\_LEASE Database Table

This table was added in WebLogic Portal version 9.2. It is used internally to maintain an exclusive lease request across any application connecting to the same database, in order to prevent multiple conflicting processes. The values in the table are transient and rows in the table should exist only as long as the lease is of value. If for some reason the row is not deleted automatically, WebLogic Portal will update any row as needed when another request with the same name occurs.

The following examples show how this lease data is used:

- A WebLogic Portal lease manager locks a resource for startup/update for a specified period of time, and a propagation is running on one node in the cluster. If another node attempts to start a propagation, the lease prevents the second process from beginning at that time.
- A WebLogic Portal lease manager uses this table to synchronize distribution of portal framework data within a cluster.

This table is used for internal processing only and is not accessible using the WebLogic Portal Administration Console.

**Table 10–10 P13N\_LEASE Table Metadata**

| Column Name     | Data Type   | Null Value | Description  | New or Changed Version |
|-----------------|-------------|------------|--|------------------------|
| LEASE_NAME      | VARCHAR(80) | Not Null   | PK – Unique name of the lease requested.   | 9.2                    |
| CREATION_DATE   | TIMESTAMP   | Not Null   | Date and time the lease was requested. The default value is the current timestamp. | 9.2                    |
| EXPIRATION_DATE | TIMESTAMP   | Not Null   | Date and time the lease will expire  | 9.2                    |
| SERVER_NAME     | VARCHAR(80) | Not Null   | Name of the server requesting the lease.   | 9.2                    |

### 10.1.1.11 Credential Service Dictionary Tables

For an entity relation diagram of the Personalization Credential Service tables, see [Figure B–2, "Personalization Credential Service Database Tables"](#).

The Credential tables include the following:

- [Section 10.1.1.12, "P13N\\_CREDENTIAL Database Table"](#)
- [Section 10.1.1.13, "P13N\\_CREDENTIAL\\_APPLICATION Database Table"](#)
- [Section 10.1.1.14, "P13N\\_CREDENTIAL\\_MASTER Database Table"](#)
- [Section 10.1.1.15, "P13N\\_CREDENTIAL\\_METADATA Database Table"](#)

### 10.1.1.12 P13N\_CREDENTIAL Database Table

This table was added in WebLogic Portal 10.2. The table stores credential user name and password information for accessing external system resources.

**Table 10–11 P13N\_CREDENTIAL Table Metadata**

| Column Name          | Data Type    | Null Value | Description   | New or Changed Version |
|----------------------|--------------|------------|---|------------------------|
| CREDENTIAL_ID        | INTEGER      | Not Null   | A unique system-generated number to use as the record identifier. | 10.2                   |
| CREDENTIAL_MASTER_ID | INTEGER      | Not Null   | FK – A Foreign Key to P13N_CREDENTIAL_MASTER                      | 10.2                   |
| USER_NAME            | VARCHAR(200) | Not Null   | The name of the user.   | 10.2                   |
| CREATION_DATE        | DATETIME     | Not Null   | Date and time the row was created.                                | 10.2                   |

**Table 10–11 (Cont.) P13N\_CREDENTIAL Table Metadata**

| Column Name          | Data Type | Null Value | Description   | New or Changed Version |
|----------------------|-----------|------------|---|------------------------|
| MODIFIED_DATE        | DATETIME  | Not Null   | Date and time the row was last modified.                          | 10.2                   |
| CREDENTIAL_PRINCIPAL | BLOB(5M)  | Null       | Encrypted login information for the external credentialed system. | 10.2                   |
| CREDENTIAL_PASSWORD  | BLOB(5M)  | Null       | Encrypted password for the external credentialed system.          | 10.2                   |

#### 10.1.1.13 P13N\_CREDENTIAL\_APPLICATION Database Table

This table was added in WebLogic Portal 10.2. The table is used to identify the enterprise and web application associated with a credentialed system.

**Table 10–12 P13N\_CREDENTIAL\_APPLICATION Table Metadata**

| Column Name               | Data Type    | Null Value | Description   | New or Changed Version |
|---------------------------|--------------|------------|---|------------------------|
| CREDENTIAL_APPLICATION_ID | INTEGER      | Not Null   | A unique system-generated number to use as the record identifier. | 10.2                   |
| ENTAPP_NAME               | VARCHAR(255) | Null       | The name of a credentialed enterprise application.                | 10.2                   |
| WEBAPP_NAME               | VARCHAR(255) | Null       | The name of a credentialed web application.                       | 10.2                   |

#### 10.1.1.14 P13N\_CREDENTIAL\_MASTER Database Table

This table was added in WebLogic Portal 10.2. The table is used to reference the credentialed system. For example, a SAP system or SAP portlets, or a Documentum or SharePoint Content Repository.



**Table 10–13 P13N\_CREDENTIAL\_MASTER Table Metadata**

| Column Name               | Data Type    | Null Value | Description  | New or Changed Version |
|---------------------------|--------------|------------|--|------------------------|
| CREDENTIAL_MASTER_ID      | INTEGER      | Not Null   | FK – A Foreign Key to P13N_CREDENTIAL_MASTER   | 10.2                   |
| CREDENTIAL_APPLICATION_ID | INTEGER      | Not Null   | A unique system-generated number to use as the record identifier.<br>FK – A Foreign Key to P13N_CREDENTIAL_APPLICATION | 10.2                   |
| RESOURCE_NAME             | VARCHAR(255) | Not Null   | The name of the resource having a credential.  | 10.2                   |
| CREDENTIAL_NAME           | VARCHAR(255) | Not Null   | The name identifying the credential or external credential system.   | 10.2                   |
| CREDENTIAL_SCOPE          | SMALLINT     | Not Null   | The type of the credential. Valid values include: 0=Resource; 1=User; 2=System.  | 10.2                   |
| DESCRIPTION               | VARCHAR(512) | Null       | Optional. Description of the credential or external credential system.   | 10.2                   |

**10.1.1.15 P13N\_CREDENTIAL\_METADATA Database Table**

This table was added in WebLogic Portal 10.2. The table is used to identify the attributes of the credential.

**Table 10–14 P13N\_CREDENTIAL\_METADATA Table Metadata**

| Column Name   | Data Type    | Null Value | Description   | New or Changed Version |
|---------------|--------------|------------|---|------------------------|
| CREDENTIAL_ID | INTEGER      | Not Null   | A unique system-generated number to use as the record identifier.<br>PK/FK – A Foreign Key to P13N_CREDENTIAL | 10.2                   |
| METADATA_NAME | VARCHAR(512) | Not Null   | PK - A name identifying the meta data value. For example: SAP URL.  | 10.2                   |
| CREATION_DATE | DATETIME     | Not Null   | Date and time the row was created.  | 10.2                   |
| MODIFIED_DATE | DATETIME     | Not Null   | Date and time the row was last modified.  | 10.2                   |
| TEXT_VALUE    | VARCHAR(512) | Not Null   | The actual value of the metadata.   | 10.2                   |

### 10.1.1.16 Job Scheduler Dictionary Tables

For an entity relation diagram of the Personalization Job Scheduler tables, see [Figure B–3, "Personalization Job Scheduler Database Tables"](#).

The Job Scheduler tables include the following:

- [Section 10.1.1.17, "P13N\\_JOB\\_DEFINITION Database Table"](#)
- [Section 10.1.1.18, "P13N\\_JOB\\_LISTENER Database Table"](#)
- [Section 10.1.1.19, "P13N\\_JOB\\_PARAMETER Database Table"](#)
- [Section 10.1.1.20, "P13N\\_JOB\\_RUN\\_HEADER Database Table"](#)
- [Section 10.1.1.21, "P13N\\_JOB\\_RUN\\_DATA Database Table"](#)

### 10.1.1.17 P13N\_JOB\_DEFINITION Database Table

This table was added in WebLogic Portal 10.2. The table is currently for internal use only and is reserved for future customer use. This table defines job run types and schedule information.

**Table 10–15 P13N\_JOB\_DEFINITION Table Metadata**

| Column Name         | Data Type    | Null Value | Description  | New or Changed Version |
|---------------------|--------------|------------|--|------------------------|
| JOB_NAME            | VARCHAR(255) | Not Null   | PK - A unique name for the job within a web application.   | 10.2                   |
| APPLICATION_NAME    | VARCHAR(100) | Not Null   | PK - The enterprise application name.  | 10.3                   |
| WEBAPP_NAME         | VARCHAR(255) | Null       | PK - The name of a web application.  | 10.2                   |
| JOB_GROUP_NAME      | VARCHAR(80)  | Null       | Optional. A user-defined group name associated with the job or group of jobs. For content publishing, for example, App init job group. | 10.2                   |
| SERVER_NAME         | VARCHAR(80)  | Null       | Optional. The name of the server where the job should run.   | 10.2                   |
| JOB_TYPE            | VARCHAR(10)  | Not Null   | The type of job.   | 10.2                   |
| IS_ENABLED          | SMALLINT     | Not Null   | A Boolean indicating that the job is enabled for execution. Default=1 (true).  | 10.2                   |
| IS_REMOVE_AFTER_RUN | SMALLINT     | Not Null   | A Boolean that indicates to the job manager that the job definition should be deleted after the job runs.                              | 10.2                   |
| CLASS_NAME          | VARCHAR(255) | Not Null   | Class name that the job represents.  | 10.2                   |
| RUN_INTERVAL        | INTEGER      | Null       | Time in minutes that the job should run or Null.   | 10.2                   |
| RUN_YEAR            | SMALLINT     | Null       | Year YYYY that the job should run or Null.   | 10.2                   |
| RUN_MONTH           | SMALLINT     | Null       | Month MM that the job should run or Null.  | 10.2                   |

**Table 10–15 (Cont.) P13N\_JOB\_DEFINITION Table Metadata**

| Column Name | Data Type    | Null Value | Description   | New or Changed Version |
|-------------|--------------|------------|---|------------------------|
| RUN_DAY     | SMALLINT     | Null       | Day DD that the job should run or Null.                         | 10.2                   |
| RUN_HOUR    | SMALLINT     | Null       | Hour that the job should run or Null. Based on a 24-hour clock. | 10.2                   |
| RUN_MINUTE  | SMALLINT     | Null       | Minute of the hour that the job should run. Defaults to 0.      | 10.2                   |
| DESCRIPTION | VARCHAR(255) | Null       | Optional. User-defined description of the job.                  | 10.2                   |

**10.1.1.18 P13N\_JOB\_LISTENER Database Table**

This table was added in WebLogic Portal 10.2. The table is currently for internal use only and is reserved for future customer use. The table defines listener classes for a given job.

**Table 10–16 P13N\_JOB\_LISTENER Table Metadata**

| Column Name      | Data Type    | Null Value | Description   | New or Changed Version |
|------------------|--------------|------------|---|------------------------|
| JOB_NAME         | VARCHAR(255) | Not Null   | PK – A unique name for the job.<br>FK - references P13N_JOB_DEFINITION.<br>JOB_NAME           | 10.2                   |
| LISTENER_NAME    | VARCHAR(255) | Not Null   | PK – A unique name for the job listener.  | 10.2                   |
| CLASS_NAME       | VARCHAR(255) | Not Null   | Job listener class name.  | 10.2                   |
| APPLICATION_NAME | VARCHAR(100) | Not Null   | PK - The enterprise application name.<br>FK - references P13N_JOB_DEFINITION.APPLICATION_NAME | 10.3                   |
| WEBAPP_NAME      | VARCHAR(255) | Not Null   | PK - The web application name.<br>FK - References P13N_JOB_DEFINITION.WEBAPP_NAME             | 10.3                   |

**10.1.1.19 P13N\_JOB\_PARAMETER Database Table**

This table was added in WebLogic Portal 10.2. The table is currently for internal use only and is reserved for future customer use. The table defines job run parameter name value pairs.

**Table 10–17 P13N\_JOB\_PARAMETER Table Metadata**

| Column Name      | Data Type    | Null Value | Description   | New or Changed Version |
|------------------|--------------|------------|---|------------------------|
| JOB_NAME         | VARCHAR(255) | Not Null   | PK – A unique name for the job.<br>FK – References P13N_JOB_DEFINITION.<br>JOB_NAME           | 10.2                   |
| APPLICATION_NAME | VARCHAR(100) | Not Null   | PK - The enterprise application name.<br>FK - references P13N_JOB_DEFINITION.APPLICATION_NAME | 10.3                   |
| WEBAPP_NAME      | VARCHAR(255) | Not Null   | PK - The web application name<br>FK - References P13N_JOB_DEFINITION.WEBAPP_NAME              | 10.3                   |
| PARAMETER_NAME   | VARCHAR(255) | Not Null   | PK – A unique job parameter name supplied at job run time. For example, App default e-mail.   | 10.2                   |
| PARAMETER_VALUE  | VARCHAR(255) | Not Null   | A job parameter value supplied at run time. For example, feedback@customer.com.               | 10.2                   |

**10.1.1.20 P13N\_JOB\_RUN\_HEADER Database Table**

This table was added in WebLogic Portal 10.2. The table is currently for internal use only and is reserved for future customer use. The table groups run information about job executions.

**Table 10–18 P13N\_JOB\_RUN\_HEADER Table Metadata**

| Column Name       | Data Type    | Null Value | Description  | New or Changed Version |
|-------------------|--------------|------------|--|------------------------|
| JOB_RUN_HEADER_ID | INTEGER      | Not Null   | PK – A unique system-generated header ID for the job run.    | 10.2                   |
| JOB_NAME          | VARCHAR(255) | Not Null   | A unique name for the job.                                   | 10.2                   |
| SERVER_NAME       | VARCHAR(80)  | Not Null   | Optional. The name of the server where the job should run.   | 10.2                   |
| APPLICATION_NAME  | VARCHAR(100) | Not Null   | The application name.  | 10.3                   |
| WEBAPP_NAME       | VARCHAR(255) | Null       | The name of the web application in which the job should run. | 10.2                   |

### 10.1.1.21 P13N\_JOB\_RUN\_DATA Database Table

This table was added in WebLogic Portal 10.2. The table is currently for internal use only and is reserved for future customer use. The table stores job run data associated with execution of the job.

**Table 10–19 P13N\_JOB\_RUN\_DATA Table Metadata**

| Column Name       | Data Type     | Null Value | Description  | New or Changed Version |
|-------------------|---------------|------------|--|------------------------|
| JOB_RUN_GUID      | VARCHAR(255)  | Not Null   | PK – A system-generated, global, unique identifier for the job's execution.                      | 10.2                   |
| JOB_RUN_HEADER_ID | INTEGER       | Not Null   | FK – A Foreign Key to P13N_JOB_RUN_HEADER.   | 10.2                   |
| RUN_STATUS        | VARCHAR(20)   | Not Null   | Status of the job run. Valid values include RUNNING, COMPLETED, INCOMPLETE, FAILED, and SUCCESS. | 10.2                   |
| START_DATETIME    | NUMERIC(15,0) | Null       | The date and time the job run started.   | 10.2                   |
| END_DATETIME      | NUMERIC(15,0) | Null       | The date and time the job run finished.  | 10.2                   |
| LEASE_NAME        | VARCHAR(160)  | Null       | The name of the lease associated with the job run.   | 10.2                   |
| RUN_DATA          | CLOB(30M)     | Null       | Data retained about the job run.   | 10.2                   |
| ERROR_DATA        | CLOB(30M)     | Null       | Error information for a failed job.  | 10.2                   |

### 10.1.1.22 Entitlement Reference Dictionary Tables

For an entity relation diagram of the Entitlement database tables, see [Figure B–4, "Entitlement Database Tables"](#).

The following tables are used by the Administration Console 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 tables include the following:

- [Section 10.1.1.23, "P13N\\_ENTITLEMENT\\_APPLICATION Database Table"](#)
- [Section 10.1.1.24, "P13N\\_ENTITLEMENT\\_POLICY Database Table"](#)
- [Section 10.1.1.25, "P13N\\_ENTITLEMENT\\_RESOURCE Database Table"](#)
- [Section 10.1.1.26, "P13N\\_ENTITLEMENT\\_ROLE Database Table"](#)
- [Section 10.1.1.27, "P13N\\_DELEGATED\\_HIERARCHY Database Table"](#)

### 10.1.1.23 P13N\_ENTITLEMENT\_APPLICATION Database Table

This table uniquely identifies an application for which entitlements can be applied.

**Table 10–20 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. |

#### 10.1.1.24 P13N\_ENTITLEMENT\_POLICY Database Table

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

**Table 10–21 P13N\_ENTITLEMENT\_POLICY Table Metadata**

| Column Name                | Data Type   | Null Value | Description  | New or Changed Version |
|----------------------------|-------------|------------|--|------------------------|
| 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; default value is the current time stamp. |                        |
| ROLE_PROVIDER_NAME         | VARCHAR(80) | Not Null   | The name given the deployed Role Mapping provider holding the role definition.           |                        |
| ATZ_PROVIDER_NAME          | VARCHAR(80) | Not Null   | The name of the authorization provider.  | 9.2                    |

**10.1.1.25 P13N\_ENTITLEMENT\_RESOURCE Database Table**

This table uniquely identifies an application resource that can have an entitlement associated with it.

**Table 10–22 P13N\_ENTITLEMENT\_RESOURCE Table Metadata**

| Column Name          | Data Type    | Null Value | Description  | New or Changed Version |
|----------------------|--------------|------------|--|------------------------|
| 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(882) | Not Null   | The name of the resource having a policy applied on it.                                | 9.2                    |
| 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. |                        |

**10.1.1.26 P13N\_ENTITLEMENT\_ROLE Database Table**

This table uniquely identifies entitlement and delegated administration roles for a given application.

**Table 10–23 P13N\_ENTITLEMENT\_ROLE Table Metadata**

| Column Name        | Data Type    | Null Value | Description   | New or Changed Version |
|--------------------|--------------|------------|---|------------------------|
| 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.                                  |                        |

**Table 10–23 (Cont.) P13N\_ENTITLEMENT\_ROLE Table Metadata**

| Column Name        | Data Type   | Null Value | Description  | New or Changed Version |
|--------------------|-------------|------------|--|------------------------|
| 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. |                        |
| ROLE_PROVIDER_NAME | VARCHAR(80) | Not Null   | Identifies the policy role mapper provider; default value = DefaultRoleMapper          | 9.2                    |

### 10.1.1.27 P13N\_DELEGATED\_HIERARCHY Database Table

This table uniquely identifies an entitlement hierarchy. An entitlement hierarchy is associated with P13N\_ENTITLEMENT\_APPLICATION.

**Table 10–24 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. |

## 10.1.2 Tracked Anonymous User Dictionary Tables

This section documents the database objects for the WebLogic Portal package. For more information about anonymous users, refer to the *Oracle Fusion Middleware User Management Guide for Oracle WebLogic Portal*.

The following tables support tracking of anonymous users:

- [Section 10.1.2.1, "P13N\\_ANONYMOUS\\_PROPERTY Database Table"](#)
- [Section 10.1.2.2, "P13N\\_ANONYMOUS\\_USER Database Table"](#)

### 10.1.2.1 P13N\_ANONYMOUS\_PROPERTY Database Table

This table stores the properties associated with the tracked anonymous user.



**Table 10–25 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   | FK - A foreign key to P13N_ANONYMOUS_USER.  |
| PROPERTY_VALUE        | BLOB         | Not Null   | The value of the property that must implement <code>java.io.Serializable</code> .                       |

### 10.1.2.2 P13N\_ANONYMOUS\_USER Database Table

This table stores the tracked anonymous user data.

**Table 10–26 P13N\_ANONYMOUS\_USER Table Metadata**

| Column Name       | Data Type    | Null Value | Description   |
|-------------------|--------------|------------|---|
| ANONYMOUS_USER_ID | VARCHAR(128) | Not Null   | FK – Maps to the primary key of the anonymous 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. |
| LAST_VISIT_DATE   | DATE         | Null       | Date when the tracked anonymous user last updated the data.   |

## 10.1.3 Behavior Tracking Database Tables

For an entity relation diagram of the Behavior Tracking tables, see [Figure B–5, "Behavior Tracking Database Tables"](#).

Each DBMS-specific chapter of this guide describes how to set up a separate behavior tracking database. Three tables are provided for 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.

For more information about behavior tracking, refer to the *Oracle Fusion Middleware Interaction Management Guide for Oracle WebLogic Portal*.

- [Section 10.1.3.1, "BT\\_EVENT\\_TYPE Database Table"](#)
- [Section 10.1.3.2, "BT\\_EVENT Database Table"](#)
- [Section 10.1.3.3, "BT\\_EVENT\\_ACTION Database Table"](#)

### 10.1.3.1 BT\_EVENT\_TYPE Database Table

This table references event types and categories in the BT\_EVENT table. This table is static.

**Table 10–27** BT\_EVENT\_TYPE Database Table

| 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.

### 10.1.3.2 BT\_EVENT Database Table

This table stores all behavior tracking event data.

**Table 10–28** BT\_EVENT Database Table

| 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 <a href="#">Table 10–42</a> . |
| 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 10–42](#), the BT\_EVENT table has eight columns; each column corresponds to a specific event element. Seven of the BT\_EVENT table's columns contain data common to every event type. The XML\_DEFINITION column contains information about the application, event-date, event-type, session-id, and user-id, 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 10–42](#).

**Table 10–29 XML\_DEFINITION Data Elements**

| <b>Event</b>              | <b>Data Element</b>  |
|---------------------------|--|
| AddToCartEvent            | application<br>event-date<br>event-type<br>session-id<br>user-id<br>sku<br>quantity<br>unit-list-price<br>currency<br>application-name                               |
| BuyEvent                  | application<br>event-date<br>event-type<br>session-id<br>user-id<br>sku<br>quantity<br>unit-price<br>currency<br>application-name<br>order-line-id                   |
| CampaignUserActivityEvent | application<br>event-date<br>event-type<br>session-id<br>user-id<br>campaign-id<br>scenario-id   |
| ClickCampaignEvent        | application<br>event-date<br>event-type<br>session-id<br>user-id<br>document-type<br>document-id<br>campaign-id<br>scenario-id<br>application-name<br>placeholder-id |
| ClickContentEvent         | application<br>event-date<br>event-type<br>session-id<br>user-id<br>document-type<br>document-id   |

**Table 10–29 (Cont.) XML\_DEFINITION Data Elements**

| <b>Event</b>         | <b>Data Element</b>  |
|----------------------|--|
| ClickProductEvent    | application<br>event-date<br>event-type<br>session-id<br>user-id<br>document-type<br>document-id<br>sku<br>category-id<br>application-name                           |
| DisplayCampaignEvent | application<br>event-date<br>event-type<br>session-id<br>user-id<br>document-type<br>document-id<br>campaign-id<br>scenario-id<br>application-name<br>placeholder-id |
| DisplayContentEvent  | application<br>event-date<br>event-type<br>session-id<br>user-id<br>document-type<br>document-id   |
| DisplayProductEvent  | application<br>event-date<br>event-type<br>session-id<br>user-id<br>document-type<br>document-id<br>sku<br>category-id<br>application-name                           |
| PurchaseCartEvent    | application<br>event-date<br>event-type<br>session-id<br>user-id<br>total-price<br>order-id<br>currency<br>application-name  |

**Table 10–29 (Cont.) XML\_DEFINITION Data Elements**

| <b>Event</b>          | <b>Data Element</b>   |
|-----------------------|---|
| RemoveFromCartEvent   | application<br>event-date<br>event-type<br>session-id<br>user-id<br>sku<br>quantity<br>unit-price<br>currency<br>application-name |
| RuleEvent             | application<br>event-date<br>event-type<br>session-id<br>user-id<br>ruleset-name<br>rule-name                                     |
| SessionBeginEvent     | application<br>event-date<br>event-type<br>session-id<br>user-id  |
| SessionEndEvent       | application<br>event-date<br>event-type<br>session-id<br>user-id  |
| SessionLoginEvent     | application<br>event-date<br>event-type<br>session-id<br>user-id  |
| UserRegistrationEvent | application<br>event-date<br>event-type<br>session-id<br>user-id  |

### 10.1.3.3 BT\_EVENT\_ACTION Database Table

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

**Table 10–30 BT\_EVENT\_ACTION Table Metadata**

| <b>Column Name</b> | <b>Data Type</b> | <b>Null Value</b> | <b>Description</b>  |
|--------------------|------------------|-------------------|---|
| 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.   |

## 10.1.4 Data Synchronization Database Tables

For an entity relation diagram of the Data Synchronization tables, see [Figure B-6, "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:

- [Section 10.1.4.1, "DATA\\_SYNC\\_APPLICATION Database Table"](#)
- [Section 10.1.4.2, "DATA\\_SYNC\\_ITEM Database Table"](#)
- [Section 10.1.4.3, "DATA\\_SYNC\\_SCHEMA\\_URI Database Table"](#)
- [Section 10.1.4.4, "DATA\\_SYNC\\_VERSION Database Table"](#)

### 10.1.4.1 DATA\_SYNC\_APPLICATION Database Table

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

**Table 10-31 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.   |

### 10.1.4.2 DATA\_SYNC\_ITEM Database Table

This table stores all the data items to be synchronized.

**Table 10-32 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_APPLICATION.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 10–32 (Cont.) DATA\_SYNC\_ITEM Table Metadata**

| 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 information – the OS login.   |
| ITEM_NAME         | VARCHAR(100) | Null       | Metadata information – the full path to the item.                            |
| ITEM_DESCRIPTION  | VARCHAR(254) | Null       | Metadata information – a general description of the item to be synchronized. |

#### 10.1.4.3 DATA\_SYNC\_SCHEMA\_URI Database Table

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

**Table 10–33 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.                      |

#### 10.1.4.4 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 10–34 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.        |

## 10.2 Portal Services Database Objects

Portal Services database objects are used for placeholders, ads, and ad bucket services.

For an entity relation diagram of the Portal Services database tables, see [Figure B-7, "Portal Services Database Tables"](#).

The following tables compose the Portal Services database:

- [Section 10.2.1, "AD\\_BUCKET Database Table"](#)
- [Section 10.2.2, "AD\\_COUNT Database Table"](#)
- [Section 10.2.3, "PLACEHOLDER\\_PREVIEW Database Table"](#)
- [Section 10.2.4, "MAIL\\_ADDRESS Database Table"](#)
- [Section 10.2.5, "MAIL\\_BATCH Database Table"](#)
- [Section 10.2.6, "MAIL\\_BATCH\\_ENTRY Database Table"](#)
- [Section 10.2.7, "MAIL\\_HEADER Database Table"](#)
- [Section 10.2.8, "MAIL\\_MESSAGE Database Table"](#)
- [Section 10.2.9, "SCENARIO\\_END\\_STATE Database Table"](#)

### 10.2.1 AD\_BUCKET Database Table

This table maintains content queries for ads.

**Table 10-35 AD\_BUCKET Table Metadata**

| Column Name         | Data Type     | Null Value | Description   |
|---------------------|---------------|------------|---|
| AD_BUCKET_ID        | NUMERIC(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.             |
| WEIGHT              | NUMERIC(15)   | Null       | A weighted scheme used in prioritizing one placeholder over another.  |
| VIEW_COUNT          | NUMERIC(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.   |



## 10.2.2 AD\_COUNT Database Table

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

**Table 10–36 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       | NUMERIC(15)  | Not Null   | The number of times the ad has been displayed.                                 |
| CLICK_THROUGH_COUNT | NUMERIC(15)  | Not Null   | The number of times the ad has been clicked.                                   |

## 10.2.3 PLACEHOLDER\_PREVIEW Database Table

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

**Table 10–37 PLACEHOLDER\_PREVIEW Table Metadata**

| Column Name    | Data Type   | Null Value | Description   |
|----------------|-------------|------------|---|
| PREVIEW_ID     | NUMERIC(15) | 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.                 |

## 10.2.4 MAIL\_ADDRESS Database Table

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

**Table 10–38 MAIL\_ADDRESS Table Metadata**

| Column Name     | Data Type    | Null Value | Description  |
|-----------------|--------------|------------|--|
| MAIL_ADDRESS_ID | NUMERIC(15)  | Not Null   | PK – A unique, system-generated number to use as the record ID.  |
| MESSAGE_ID      | NUMERIC(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. |

## 10.2.5 MAIL\_BATCH Database Table

This table establishes a batch for each mailing.

**Table 10–39 MAIL\_BATCH Table Metadata**

| Column Name | Data Type    | Null Value | Description   |
|-------------|--------------|------------|---|
| BATCH_ID    | NUMERIC(15)  | Not Null   | PK – A unique, system-generated number to use as the record ID. |
| BATCH_NAME  | VARCHAR(254) | Not Null   | The name of the mail message batch.                             |

## 10.2.6 MAIL\_BATCH\_ENTRY Database Table

This table correlates the mail batch with the specific mail message.

**Table 10–40 MAIL\_BATCH\_ENTRY Table Metadata**

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

## 10.2.7 MAIL\_HEADER Database Table

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

**Table 10–41 MAIL\_HEADER Table Metadata**

| Column Name  | Data Type    | Null Value | Description   |
|--------------|--------------|------------|---|
| HEADER_ID    | NUMERIC(15)  | Not Null   | PK – A unique, system-generated number to use as the record ID. |
| MESSAGE_ID   | NUMERIC(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.                           |

## 10.2.8 MAIL\_MESSAGE Database Table

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

**Table 10–42 MAIL\_MESSAGE Table Metadata**

| Column Name  | Data Type    | Null Value | Description   |
|--------------|--------------|------------|---|
| MESSAGE_ID   | NUMERIC(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.                          |

## 10.2.9 SCENARIO\_END\_STATE Database Table

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

**Table 10–43 SCENARIO\_END\_STATE Table Metadata**

| Column Name      | Data Type    | Null Value | Description   |
|------------------|--------------|------------|---|
| SCENARIO_XML_REF | VARCHAR(254) | Not Null   | PK – The identifier for the XML-based scenario definition file.                                     |
| USER_NAME        | VARCHAR(200) | Not Null   | PK—the user ID.   |
| 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. |

## 10.3 Portal Framework Database Objects

The Portal Framework is the portion of WebLogic Portal that is responsible for the rendering and customization of the portal.

For an entity relation diagram of the Portal Framework database tables, see [Figure B–8, "Portal Framework Database Tables"](#). This includes tables that support WSRP and localization.

### 10.3.1 Portal Framework Database Tables

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

- [Section 10.3.1.1, "PF\\_BOOK\\_DEFINITION Database Table"](#)
- [Section 10.3.1.2, "PF\\_BOOK\\_GROUP Database Table"](#)
- [Section 10.3.1.3, "PF\\_BOOK\\_INSTANCE Database Table"](#)
- [Section 10.3.1.4, "PF\\_DESKTOP\\_DEFINITION Database Table"](#)
- [Section 10.3.1.5, "PF\\_DESKTOP\\_INSTANCE Database Table"](#)
- [Section 10.3.1.6, "PF\\_LAYOUT\\_DEFINITION Database Table"](#)
- [Section 10.3.1.7, "PF\\_LOOK\\_AND\\_FEEL\\_DEFINITION Database Table"](#)
- [Section 10.3.1.8, "PF\\_MARKUP\\_DEFINITION Database Table"](#)
- [Section 10.3.1.9, "PF\\_MARKUP\\_XML Database Table"](#)
- [Section 10.3.1.11, "PF\\_PAGE\\_DEFINITION Database Table"](#)
- [Section 10.3.1.12, "PF\\_PAGE\\_INSTANCE Database Table"](#)
- [Section 10.3.1.13, "PF\\_PLACEHOLDER\\_DEFINITION Database Table"](#)
- [Section 10.3.1.14, "PF\\_PLACEMENT Database Table"](#)
- [Section 10.3.1.15, "PF\\_PORTAL Database Table"](#)
- [Section 10.3.1.16, "PF\\_PORTLET\\_CATEGORY Database Table"](#)
- [Section 10.3.1.17, "PF\\_PORTLET\\_CATEGORY\\_DEFINITION Database Table"](#)
- [Section 10.3.1.18, "PF\\_PORTLET\\_DEFINITION Database Table"](#)
- [Section 10.3.1.19, "PF\\_PORTLET\\_INSTANCE Database Table"](#)

- [Section 10.3.1.20, "PF\\_PORTLET\\_PREFERENCE Database Table"](#)
- [Section 10.3.1.21, "PF\\_PORTLET\\_PREFERENCE\\_VALUE Database Table"](#)
- [Section 10.3.1.22, "PF\\_PORTLET\\_USER\\_PROPERTIES Database Table"](#)
- [Section 10.3.1.23, "PF\\_SHELL\\_DEFINITION Database Table"](#)
- [Section 10.3.1.24, "PF\\_SHELL\\_PLACEMENT Database Table"](#)
- [Section 10.3.1.25, "PF\\_THEME\\_DEFINITION Database Table"](#)
- [Section 10.3.1.26, "PF\\_URL\\_COMPRESSION Database Table"](#)

To view the communities-related portal framework tables, see [Section 10.5, "Communities Framework Database Objects."](#) To view the tables related to localization, refer to [Section 10.3.2, "Localization Dictionary Tables."](#)

### 10.3.1.1 PF\_BOOK\_DEFINITION Database Table

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

**Table 10–44 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.<br><br>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 <code>config.xml</code> ) 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 <code>bookDefinitionLabel</code> in Oracle Enterprise Pack for Eclipse.<br><br>If a label is not supplied at creation time, the <code>BOOK_DEFINITION_ID</code> prefixed with a B is used. This label can be supplied to APIs to activate books or pages. |

### 10.3.1.2 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 10–45 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.  |
| 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. |

### 10.3.1.3 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 10–46 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.  |
| 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.   |

#### 10.3.1.4 PF\_DESKTOP\_DEFINITION Database Table

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

**Table 10–47 PF\_DESKTOP\_DEFINITION Table Metadata**

| Column Name   | Data Type   | Null Value | Description  | New or Changed Version |
|---------------|-------------|------------|--|------------------------|
| 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 web application (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.   |                        |

**Table 10–47 (Cont.) PF\_DESKTOP\_DEFINITION Table Metadata**

| Column Name                 | Data Type   | Null Value | Description   | New or Changed Version |
|-----------------------------|-------------|------------|---|------------------------|
| 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. 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).  |                        |
| DESKTOP_TYPE                | NUMBER      | Not Null   | Indicates whether the desktop is a community; the default is 0 (not a community).   |                        |
| IS_TEMPLATE                 | NUMBER      | Not Null   | Indicates whether the desktop is a template; the default is 0 (not a template).   |                        |
| IS_GLOBAL                   | NUMBER      | Not Null   | Indicates whether the desktop is a global desktop; the default is 0 (not global).   |                        |
| ASYNC_MODE_TYPE             | VARCHAR(10) | Not Null   | Adds the ability to turn running a desktop in AJAX on or off at the desktop level. The default is <code>compat_9_2</code> , or off.                               | 10.0                   |
| IS_SCROLL_TO_WINDOW_ENABLED | SMALLINT    | Not Null   | A Boolean flag that enables scrolling of the browser window to the last window where the user interacted. The default is 0 (true).                                | 10.0                   |
| IS_DVT_ENABLED              | SMALLINT    | Not Null   | A flag that indicates if the desktop has client-side dynamic visitor tooling enabled. Default is 0 (false).   | 10.2                   |
| IS_DISC_ENABLED             | SMALLINT    | Not Null   | A flag that indicates if the desktop has dynamic interface scripting context enabled for client-side and server-wide framework integration. Default is 0 (false). | 10.2                   |

### 10.3.1.5 PF\_DESKTOP\_INSTANCE Database Table

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

**Table 10–48 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.  |
| 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.<br><br>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.  |

### 10.3.1.6 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 10–49 PF\_LAYOUT\_DEFINITION Table Metadata**

| Column Name          | Data Type | Null Value | Description   | New or Changed Version |
|----------------------|-----------|------------|---|------------------------|
| 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.                          |                        |



**Table 10–49 (Cont.) PF\_LAYOUT\_DEFINITION Table Metadata**

| Column Name            | Data Type    | Null Value | Description  | New or Changed Version |
|------------------------|--------------|------------|--|------------------------|
| 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_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.<br><br>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.   |                        |
| THUMBNAIL_URI          | VARCHAR(255) | Null       | Associates a thumbnail image with the layout for display in the Administration Console or Visitor Tools.   | 10.0                   |

### 10.3.1.7 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 10–50 PF\_LOOK\_AND\_FEEL\_DEFINITION Table Metadata**

| Column Name               | Data Type    | Null Value | Description  | New or Changed Version |
|---------------------------|--------------|------------|--|------------------------|
| 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.   |                        |
| 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.<br><br>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 Administration Console. |                        |
| 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.   |                        |
| THUMBNAIL_URI             | VARCHAR(255) | Null       | Associates a thumbnail image with the look and feel for display in the Administration Console or Visitor Tools.  | 10.0                   |

### 10.3.1.8 PF\_MARKUP\_DEFINITION Database Table

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

**Table 10–51 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.   |
| 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 filename 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.                                |

### 10.3.1.9 PF\_MARKUP\_XML Database Table

This table was added in WebLogic Portal version 9.2. It defines additional XML markup information for portal resources.

**Table 10–52 PF\_MARKUP\_XML Table Metadata**

| Column Name          | Data Type     | Null Value | Description  | New or Changed Version |
|----------------------|---------------|------------|--|------------------------|
| MARKUP_DEFINITION_ID | INTEGER       | Not Null   | PK – A unique, system-generated number to use as the record ID.                            | 9.2                    |
| XML_TYPE             | CHAR(1)       | Not Null   | Identifies if this block is a begin or end block. Values are "B" for begin or "E" for end. | 9.2                    |
| BLOCK_NUMBER         | INTEGER       | Not Null   | Determines the order of the markup.  | 9.2                    |
| MARKUP_XML           | VARCHAR(2000) | Not Null   | The markup value.  | 9.2                    |

### 10.3.1.10 PF\_MENU\_DEFINITION Database Table

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

**Table 10–53 PF\_MENU\_DEFINITION Table Metadata**

| Column Name          | Data Type    | Null Value | Description  | New or Changed Version |
|----------------------|--------------|------------|--|------------------------|
| 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.   |                        |
| 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.<br><br>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 Administration Console. |                        |
| 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.  |                        |
| THUMBNAIL_URI        | VARCHAR(255) | Null       | Associates a thumbnail image with the menu for display in the Administration Console or Visitor Tools.   | 10.0                   |

### 10.3.1.11 PF\_PAGE\_DEFINITION Database Table

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

**Table 10–54 PF\_PAGE\_DEFINITION Table Metadata**

| Column Name          | Data Type | Null Value | Description  | New or Changed Version |
|----------------------|-----------|------------|--|------------------------|
| 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.  |                        |
| IS_PUBLIC            | NUMBER    | Not Null   | A boolean indicating this page definition is public. Only public page definitions are ever exposed to site visitors. |                        |

**Table 10–54 (Cont.) PF\_PAGE\_DEFINITION Table Metadata**

| Column Name     | Data Type   | Null Value | Description  |
|-----------------|-------------|------------|--|
| 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.   |

### 10.3.1.12 PF\_PAGE\_INSTANCE Database Table

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

**Table 10–55 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.                                   |
| 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. |

### 10.3.1.13 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 10–56 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.  |
| 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.   |

#### 10.3.1.14 PF\_PLACEMENT Database Table

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

**Table 10–57 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.   |
| 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. |

#### 10.3.1.15 PF\_PORTAL Database Table

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

**Table 10–58 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. |

### 10.3.1.16 PF\_PORTLET\_CATEGORY Database Table

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

**Table 10–59 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. |

### 10.3.1.17 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 10–60 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. |

**Table 10–60 (Cont.) PF\_PORTLET\_CATEGORY\_DEFINITION Table Metadata**

| Column Name                   | Data Type   | Null Value | Description   |
|-------------------------------|-------------|------------|---|
| 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. |

### 10.3.1.18 PF\_PORTLET\_DEFINITION Database Table

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

**Table 10–61 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 site 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.<br><br>Can have a value other than -1 only if IS_CACHEABLE is true.   |
| WSRP_PROPERTIES_MODE  | NUMBER    | Not Null   | Indicates required user properties for WSRP user profile propagation. Possible values are:<br>0 = None<br>1 = All<br>2 = Specified properties<br>3 = User properties are defaulted to the web application's default. This is the default value. |



**Table 10–61 (Cont.) PF\_PORTLET\_DEFINITION Table Metadata**

| 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.<br><br>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 Administration Console. |
| 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).<br><br>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.   |
| PORTLET_FILE            | VARCHAR(255) | Null       | The (* .portlet) file describing the controls that make up the portlet. The field is null if no portlet file exists.  |

**10.3.1.19 PF\_PORTLET\_INSTANCE Database Table**

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

---

**Note:** To view the WSRP tables that are a subset of the PF\_PORTLET\_INSTANCE table, see [Section 10.4, "WSRP Portal Framework Database Objects."](#)

---

**Table 10–62 PF\_PORTLET\_INSTANCE Table Metadata**

| Column Name                | Data Type   | Null Value | Description   | New or Changed Version |
|----------------------------|-------------|------------|---|------------------------|
| 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, 2=Library, 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.   |                        |
| TITLE_KEY                  | VARCHAR(80) | Null       | The bundle key for the localized title of the portlet for the bundle contained in the .portlet file.  | 10.0                   |

### 10.3.1.20 PF\_PORTLET\_PREFERENCE Database Table

This table identifies preference values for the portlet instance.

**Table 10–63 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.   |
| PREFERENCE_NAME        | VARCHAR(80)  | 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.   |

### 10.3.1.21 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 10–64 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(80)  | 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.   |

### 10.3.1.22 PF\_PORTLET\_USER\_PROPERTIES Database Table

This table maintains values of portlet user properties for WSRP user profile propagation.

This table was added in WebLogic Portal 9.2.

**Table 10–65 PF\_PORTLET\_USER\_PROPERTIES Table Metadata**

| Column Name           | Data Type    | Null Value | Description   |
|-----------------------|--------------|------------|---|
| USER_PROPERTY_ID      | NUMBER       | Not Null   | PK – A unique, system-generated number to use as the property ID.           |
| PORTLET_DEFINITION_ID | NUMBER       | Not Null   | PK – A unique, system-generated number to use as the portlet definition ID. |
| PROPERTY_STRING       | VARCHAR(255) | Not Null   | String value.   |

### 10.3.1.23 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 10–66 PF\_SHELL\_DEFINITION Table Metadata**

| Column Name           | Data Type     | Null Value | Description  | New or Changed Version |
|-----------------------|---------------|------------|--|------------------------|
| 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.<br><br>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 Administration Console. |                        |
| 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.   |                        |

**Table 10–66 (Cont.) PF\_SHELL\_DEFINITION Table Metadata**

| Column Name                 | Data Type     | Null Value | Description  | New or Changed Version |
|-----------------------------|---------------|------------|--|------------------------|
| THUMBNAIL_URI               | VARCHAR (255) | Null       | Associate a thumbnail image with the shell for display in the Administration Console or Visitor Tools. | 10.0                   |
| HEADER_LAYOUT_DEFINITION_ID | INTEGER       | Null       | FK to PF_LAYOUT_DEFINITION.  | 10.0                   |
| FOOTER_LAYOUT_DEFINITION_ID | INTEGER       | Null       | FK to PF_LAYOUT_DEFINITION.  | 10.0                   |

### 10.3.1.24 PF\_SHELL\_PLACEMENT Database Table

This table supports disassembling layouts in the `.shell` files. Each record represents the item (either book or portlet) placement in the layout for a header or footer in the `.shell` file.

**Table 10–67 PF\_SHELL\_PLACEMENT Table Metadata**

| Column Name               | Data Type                                 | Null Value | Description  | New or Changed Version |
|---------------------------|---|------------|--|------------------------|
| PLACEMENT_ID              | INTEGER                                   | Not Null   | PK – A unique, system-generated number to use as the record ID.  | 10.0                   |
| SHELL_DEFINITION_ID       | INTEGER                                   | Not Null   | FK to PF_SHELL_DEFINITION on DELETE CASCADE.   | 10.0                   |
| IS_IN_HEADER              | SMALLINT                                  | Not Null   | A Boolean indicating that the shell placement is defined for the header of the book or portlet.                  | 10.0                   |
| POSITION                  | SMALLINT                                  | Not Null   | The position within the placeholder where this placement lies. Placeholders can contain more than one placement. | 10.0                   |
| CREATION_DATE             | TIMESTAMP<br>DEFAULT<br>CURRENT_TIMESTAMP | Not Null   | The date and time the row was created.   | 10.0                   |
| MODIFIED_DATE             | TIMESTAMP<br>DEFAULT<br>CURRENT_TIMESTAMP | Not Null   | The date and time that the record was last modified.   | 10.0                   |
| PLACEHOLDER_DEFINITION_ID | INTEGER                                   | Not Null   | FK to PF_PLACEHOLDER_DEFINITION.   | 10.0                   |
| PORTLET_INSTANCE_ID       | INTEGER                                   | Null       | FK to PF_PORTLET_INSTANCE.   | 10.0                   |
| BOOK_INSTANCE_ID          | INTEGER                                   | Null       | FK to PF_BOOK_INSTANCE.  | 10.0                   |

**10.3.1.25 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 10–68 PF\_THEME\_DEFINITION Table Metadata**

| Column Name           | Data Type    | Null Value | Description  | New or Changed Version |
|-----------------------|--------------|------------|--|------------------------|
| 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.<br><br>This flag is set to true only when the <code>.theme</code> file is deleted and the theme is still in use. You can either return the <code>.theme</code> file and this flag is automatically reset, or remove the theme using the Administration Console. |                        |
| THEME_FILE            | VARCHAR(255) | Not Null   | The name of the <code>.theme</code> file contained in the application's <code>framework/markup/theme</code> directory backing this theme definition.   |                        |
| THUMBNAIL_URI         | VARCHAR(255) | Null       | Associates a thumbnail image with the theme for display in the Administration Console or Visitor Tools.  | 10.0                   |

### 10.3.1.26 PF\_URL\_COMPRESSION Database Table

This table was added in WebLogic Portal version 9.2. It maintains URL compression values.

**Table 10–69 PF\_URL\_COMPRESSION Table Metadata**

| Column Name        | Data Type     | Null Value | Description  | New or Changed Version |
|--------------------|---------------|------------|--|------------------------|
| URL_COMPRESSION_ID | INTEGER       | Not Null   | PK – A unique, system-generated number to use as the URL Compression ID.   | 9.2                    |
| EXPANDED_URL_HASH  | INTEGER       | Not Null   | Hashed URL used to look up the compressed URL; hashes are not guaranteed to be unique, so a secondary search is performed on EXPANDED_URL. | 9.2                    |
| EXPANDED_URL       | VARCHAR(1000) | Not Null   | The full uncompressed URL value.   | 9.2                    |
| PAGE_LABEL         | VARCHAR(80)   | Null       | A name used to reference this portal resource for development purposes.  | 9.2                    |

## 10.3.2 Localization Dictionary Tables

This section documents the database objects for the WebLogic Portal framework. The following tables support localization:

- [Section 10.3.2.1, "L10N\\_INTERSECTION Database Table"](#)
- [Section 10.3.2.2, "L10N\\_LOCALE Database Table"](#)
- [Section 10.3.2.3, "L10N\\_RESOURCE Database Table"](#)
- [Section 10.3.2.4, "L10N\\_RESOURCE\\_TYPE Database Table"](#)

### 10.3.2.1 L10N\_INTERSECTION Database Table

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

**Table 10–70 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.   |

### 10.3.2.2 L10N\_LOCALE Database Table

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

**Table 10–71 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.             |
| VARIANT       | VARCHAR(40) | Null       | Vendor- and browser-specific code variant code that is used by the locale; for example, WIN, MAC, UNIX. |

### 10.3.2.3 L10N\_RESOURCE Database Table

This table defines the localized title and description of a localized resource.

**Table 10–72 L10N\_RESOURCE Table Metadata**

| Column Name     | Data Type     | Null Value | Description  | New or Changed Version |
|-----------------|---------------|------------|--|------------------------|
| 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.   |                        |
| TITLE_CK        | VARCHAR(160)  | Null       | Collation key column for the title; an ASCII-sortable string that will sort correctly in the given language.           | 9.2                    |
| DESCRIPTION_CK  | VARCHAR(1000) | Null       | Collation key column for the description; an ASCII-sortable string that will sort correctly in the given language.     | 9.2                    |



### 10.3.2.4 L10N\_RESOURCE\_TYPE Database Table

This table defines portal resource types for localization.

**Table 10–73 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.   |
| 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. |

## 10.4 WSRP Portal Framework Database Objects

For an entity relation diagram of the WSRP Portal Framework tables, see [Figure B–9, "WSRP Portal Framework Tables"](#).

The following WSRP-related tables are a subset of the PF\_PORTLET\_INSTANCE database table:

- [Section 10.4.1, "PF\\_CONSUMER\\_PORTLETS Database Table"](#)
- [Section 10.4.2, "PF\\_CONSUMER\\_PROPERTIES Database Table"](#)
- [Section 10.4.3, "PF\\_CONSUMER\\_REGISTRY Database Table"](#)
- [Section 10.4.4, "PF\\_PRODUCER\\_REGISTRY Database Table"](#)
- [Section 10.3.1.19, "PF\\_PORTLET\\_INSTANCE Database Table"](#)
- [Section 10.4.6, "PF\\_PROXY\\_BOOK Database Table"](#)
- [Section 10.4.7, "PF\\_PROXY\\_PAGE Database Table"](#)

### 10.4.1 PF\_CONSUMER\_PORTLETS Database Table

The PF\_CONSUMER\_PORTLETS table associates consumer IDs and portlet instance IDs so that when a consumer unregisters from a producer, the producer can clean up any portlets that it created for the consumer.

**Table 10–74 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.  |

### 10.4.2 PF\_CONSUMER\_PROPERTIES Database Table

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

**Table 10–75 PF\_CONSUMER\_PROPERTIES Database Table**

| Column Name    | Data Type     | Null Value | Description   | New or Changed Version |
|----------------|---------------|------------|---|------------------------|
| PRODUCER_ID    | INTEGER       | Not Null   | PK/FK to PF_CONSUMER_REGISTRY   |                        |
| PROPERTY_NAME  | VARCHAR (255) | Not Null   | PK – The name of the property.  | 10.2                   |
| CREATION_DATE  | DATETIME      | Not Null   | The date and time the row was created.  |                        |
| MODIFIED_DATE  | DATETIME      | Not Null   | The date and time the row was modified. The column's data is maintained using a database trigger. |                        |
| PROPERTY_VALUE | VARCHAR (255) | Null       | The value associated with the property name.  | 10.2                   |
| PROPERTY_TYPE  | NUMERIC(1)    | Not Null   | The type of property being stored (string, long, double, Boolean, or datetime).                   | 10.2                   |

### 10.4.3 PF\_CONSUMER\_REGISTRY Database Table

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

**Table 10–76 PF\_CONSUMER\_REGISTRY Database Table**

| Column Name                  | Data Type    | Null Value | Description  | New or Changed Version |
|------------------------------|--------------|------------|--|------------------------|
| 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) | Null       | Optional. URL to the description service port offered by the producer.                                 | 9.2                    |
| MARKUP_PORT_URL              | VARCHAR(255) | Null       | Optional. URL to the markup service port offered by the producer.                                      | 9.2                    |
| COOKIE_PROTOCOL              | NUMBER       | Not Null   | The cookie protocol.<br>Values: 0 = None, 1 = Per User, 2 = Per Group.                                 |                        |

**Table 10–76 (Cont.) PF\_CONSUMER\_REGISTRY Database Table**

| Column Name                      | Data Type    | Null Value | Description  | New or Changed Version |
|----------------------------------|--------------|------------|--|------------------------|
| WSDL_URL                         | VARCHAR(255) | Not Null   | URL to the WSDL offered by the producer.   |                        |
| REQUIRES_REGISTRATION            | NUMBER       | 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.   |                        |
| 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.  |                        |
| IS_DISABLED                      | SMALLINT     | Not Null   | A Boolean that indicates if the corresponding Producer is enabled or disabled for the Consumer. The default is enabled, or 0.  | 10.0                   |
| IS_STORE_REGISTRATION_PROPERTIES | SMALLINT     | Not Null   | A Boolean that indicates if registration properties should be stored. Used by the propagation framework. Default is (1), true. | 10.2                   |

#### 10.4.4 PF\_PRODUCER\_REGISTRY Database Table

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

**Table 10–77 PF\_PRODUCER\_REGISTRY Database Table**

| Column Name   | Data Type   | Null Value | Description   |
|---------------|-------------|------------|---|
| CONSUMER_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 to which the portal application is scoped. |
| CREATION_DATE | DATE        | Not Null   | The date and time the row was created.                                      |

**Table 10–77 (Cont.) PF\_PRODUCER\_REGISTRY Database Table**

| Column Name    | Data Type   | Null Value | Description   |
|----------------|-------------|------------|---|
| 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 <code>productName.majorVersion.minorVersion</code> where <code>productName</code> identifies the product that the consumer installed for its deployment, and <code>majorVersion</code> and <code>minorVersion</code> are vendor-defined indications of the version of its product. |

#### 10.4.5 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 10–78 PF\_PROXY\_PORTLET\_INSTANCE Database Table**

| Column Name                 | Data Type | Null Value | Description   |
|-----------------------------|-----------|------------|---|
| PROXY_PORTLET_INSTANCE_ID   | INTEGER   | Not Null   | PK – 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      | NUMBER    | Not Null   | A boolean indicating that URL templates are required by the producer.   |
| TEMPLATES_STORED_IN_SESSION | NUMBER    | Not Null   | A boolean indicating whether the consumer should send templates with every request. The default is 1 = True.  |
| PORTLET_STATE_CHANGE        | NUMBER    | 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. Possible values include:<br>0 = Readonly (default)<br>1 = CBW(CloneBeforeWrite)<br>2 = Read/Write. |

**Table 10–78 (Cont.) 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<br>1 = True.<br><br>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.   |
| IS_PRIMARY          | NUMBER       | Not Null   | A boolean indicating whether the proxy portlet is primary or not. When multiple versions of the same proxy portlet are being used on a consumer, operations that involve either using or removing the proxy portlet will be done on the primary version (unless specified that the operation affects all versions). The defaults is 0 (False). |

#### 10.4.6 PF\_PROXY\_BOOK Database Table

This table maintains information on proxy books for the WebLogic Portal Administration Console. If a user is adding new remote resources to the portal, this data can be used to let the user know if a particular proxy book is already in the library.

This table was added in WebLogic Portal 9.2.

**Table 10–79 PF\_PROXY\_BOOK Table Metadata**

| Column Name        | Data Type   | Null Value | Description  | New or Changed Version |
|--------------------|-------------|------------|--|------------------------|
| PROXY_BOOK_ID      | INTEGER     | Not Null   | PK – A unique, system-generated number to use as the property ID.  | 9.2                    |
| PRODUCER_HANDLE    | VARCHAR(40) | Not Null   | Uniquely identifies the producer to the consumer.  | 9.2                    |
| ORIGINAL_NAME      | VARCHAR(80) | Not Null   | Original name of the book on the producer; this name is tracked in order to find duplicates.   | 9.2                    |
| BOOK_DEFINITION_ID | INTEGER     | Not Null   | FK to PF_BOOK_DEFINITION   | 9.2                    |
| CREATION_DATE      | DATE        | Not Null   | The date and time the row was created.   | 9.2                    |
| MODIFIED_DATE      | DATE        | Not Null   | The date and time the row was last modified. This column's data is maintained using a database trigger.  | 9.2                    |
| IS_PRIMARY         | NUMBER      | Not Null   | A boolean indicating whether the proxy book is primary or not. When multiple versions of the same proxy book are being used on a consumer, operations that involve either using or removing the proxy book will be done on the primary version (unless specified that the operation affects all versions). The default is 0 (False). | 9.2                    |

### 10.4.7 PF\_PROXY\_PAGE Database Table

This table maintains information on proxy pages for the WebLogic Portal Administration Console. If a user is adding new remote resources to the portal, this data can be used to let the user know if a particular proxy page is already in the library.

This table was added in WebLogic Portal 9.2.

**Table 10–80 PF\_PROXY\_PAGE Table Metadata**

| Column Name     | Data Type   | Null Value | Description  | New or Changed Version |
|-----------------|-------------|------------|--|------------------------|
| PROXY_PAGE_ID   | INTEGER     | Not Null   | PK – A unique, system-generated number to use as the property ID.                            | 9.2                    |
| PRODUCER_HANDLE | VARCHAR(40) | Not Null   | Uniquely identifies the producer to the consumer.  | 9.2                    |
| ORIGINAL_NAME   | VARCHAR(80) | Not Null   | Original name of the book on the producer; this name is tracked in order to find duplicates. | 9.2                    |

**Table 10–80 (Cont.) PF\_PROXY\_PAGE Table Metadata**

| Column Name        | Data Type | Null Value | Description  | New or Changed Version |
|--------------------|-----------|------------|--|------------------------|
| PAGE_DEFINITION_ID | INTEGER   | Not Null   | FK to PF_PAGE_DEFINITION   | 9.2                    |
| CREATION_DATE      | DATE      | Not Null   | The date and time the row was created.   | 9.2                    |
| MODIFIED_DATE      | DATE      | Not Null   | The date and time the row was last modified. This column's data is maintained using a database trigger.  | 9.2                    |
| IS_PRIMARY         | NUMBER    | Not Null   | A boolean indicating whether a proxy page is primary or not. When multiple versions of the same proxy page are being used on a consumer, operations that involve either using or removing the proxy page will be done on the primary version (unless specified that the operation affects all versions). The default is 0 (False). | 9.2                    |

## 10.5 Communities Framework Database Objects

Communities are a new feature in WebLogic Portal version 9.2. Communities are WebLogic Portal desktops that let users with common goals and interests work together in—and administer—a web-based environment. Whether for specific events, work groups, partners, or for any other groups that need to share information, communities provide a dedicated, secure, self-managed place to collaborate.

For an entity relation diagram of the Communities Portal Framework database tables, see [Figure B–10, "Communities Portal Framework Database Tables"](#).

### 10.5.1 Communities Portal Framework Data Dictionary Tables

The Communities Portal Framework system has the following tables:

- [Section 10.5.1.1, "PF\\_COMMUNITY\\_DEFINITION Database Table"](#)
- [Section 10.5.1.2, "PF\\_COMMUNITY\\_MEMBER Database Table"](#)
- [Section 10.5.1.3, "PF\\_COMMUNITY\\_MEMBERSHIP Database Table"](#)
- [Section 10.5.1.5, "PF\\_COMMUNITY\\_PROPERTY Database Table"](#)
- [Section 10.5.1.6, "PF\\_COMMUNITY\\_PROPERTY\\_VALUE Database Table"](#)
- [Section 10.5.1.7, "PF\\_FILE\\_TEMPLATE Database Table"](#)
- [Section 10.5.1.8, "PF\\_INVITATION Database Table"](#)
- [Section 10.5.1.9, "PF\\_INVITEE Database Table"](#)
- [Section 10.5.1.10, "PF\\_INVITEE\\_PROPERTY Database Table"](#)
- [Section 10.5.1.11, "PF\\_MESSAGING\\_ADDR Database Table"](#)
- [Section 10.5.1.12, "PF\\_NOTIFICATION Database Table"](#)

- [Section 10.5.1.13, "PF\\_NOTIFICATION\\_PAYLOAD Database Table"](#)

### 10.5.1.1 PF\_COMMUNITY\_DEFINITION Database Table

The PF\_COMMUNITY\_DEFINITION table contains community definition data.

**Table 10–81 PF\_COMMUNITY\_DEFINITION Database Table**

| Column Name                | Data Type   | Null Value | Description  | New or Changed Version |
|----------------------------|-------------|------------|--|------------------------|
| COMMUNITY_DEFINITION_ID    | NUMBER      | Not Null   | PK – A unique, system-generated number to use as the community definition ID.  | 9.2                    |
| INTERSECTION_ID            | NUMBER      | Not Null   | FK to L10N_RESOURCE for the localization resource.   | 9.2                    |
| DESKTOP_PATH               | VARCHAR(40) | Not Null   | FK to PF_DESKTOP_DEFINITION.   | 9.2                    |
| PORTAL_PATH                | VARCHAR(40) | Not Null   | FK to PF_DESKTOP_DEFINITION.   | 9.2                    |
| WEBAPP_NAME                | VARCHAR(80) | Not Null   | FK to PF_DESKTOP_DEFINITION. Name of the web application that hosts the desktop.   | 9.2                    |
| TIMEZONE                   | VARCHAR(80) | Not Null   | Display TimeZone identifier. This can be used by Communities applications to format dates and times in a time zone specific to a community instance. | 9.2                    |
| IS_ACTIVE                  | NUMBER      | Not Null   | Indicates whether the community is active; default = 0 (inactive).   | 9.2                    |
| IS_PUBLIC                  | NUMBER      | Not Null   | Indicates whether the community is public; default = 0 (not public).   | 9.2                    |
| IS_TEMPLATE                | NUMBER      | Not Null   | Indicates whether the community is a template; default = 0 (not a template).   | 9.2                    |
| IS_ACCESS_TRACKING_ENABLED | NUMBER      | Not Null   | Indicates whether the community has access tracking enabled; default = 0 (not enabled).  | 9.2                    |
| IS_PERSONAL_PAGES_ENABLED  | NUMBER      | Not Null   | Indicates whether the community has personal pages enabled; default = 0 (not enabled).   | 9.2                    |
| IS_SELF_REG_ENABLED        | NUMBER      | Not Null   | Indicates whether self-registration is enabled; default = 0 (not enabled).   | 9.2                    |



**Table 10–81 (Cont.) PF\_COMMUNITY\_DEFINITION Database Table**

| Column Name                    | Data Type    | Null Value | Description  | New or Changed Version |
|--------------------------------|--------------|------------|--|------------------------|
| IS_GLOBAL                      | NUMBER       | Not Null   | Indicates whether the community is global; default = 0 (not global).   | 9.2                    |
| CREATION_DATE                  | TIMESTAMP    | Not Null   | The date and time the row was created.   | 9.2                    |
| MODIFIED_DATE                  | TIMESTAMP    | Not Null   | The date and time the row was modified. This column's data is maintained using a database trigger.   | 9.2                    |
| EXPIRATION_DATE                | TIMESTAMP    | Not Null   | The date and time this community will expire.  | 9.2                    |
| PARENT_COMMUNITY_DEFINITION_ID | NUMBER       | Null       | FK - References parent community.  | 9.2                    |
| CALLBACK_CLASS_NAME            | VARCHAR(255) | Null       | The class name for the CommunityCallback implementation that is configured for this community, if one exists.  | 9.2                    |
| PRIMARY_MESSAGING_ADDR         | VARCHAR(80)  | Null       | The MessagingAddress name that identifies the default MessagingAddressType that is configured for use with this community. If this is null, the default MessagingAddressType for all community operations will be the type configured in communities-config.xml. | 9.2                    |
| SELF_REG_URI                   | VARCHAR(255) | Null       | URI pointing to the registration page for this community instance.   | 9.2                    |
| DEACTIVATED_URI                | VARCHAR(255) | Null       | URI to page when community is either deactivated or the membership of the user has been deactivated.   | 9.2                    |

**10.5.1.2 PF\_COMMUNITY\_MEMBER Database Table**

The PF\_COMMUNITY\_MEMBER table contains user information for community members.

**Table 10–82 PF\_COMMUNITY\_MEMBER Database Table**

| Column Name         | Data Type    | Null Value | Description  | New or Changed Version |
|---------------------|--------------|------------|--|------------------------|
| COMMUNITY_MEMBER_ID | NUMBER       | Not Null   | PK – A unique, system-generated number to use as the community member ID.                          | 9.2                    |
| USER_NAME           | VARCHAR(200) | Not Null   | The WebLogic Server user name for the community member.  | 9.2                    |
| MEMBER_TYPE         | NUMBER       | Not Null   | Identifies the community member as internal (1) or external (0).                                   | 9.2                    |
| MEMBER_STATUS       | NUMBER       | Not Null   | Setting for member's community status. Values are: 0 = Disabled, 1 = Active.                       | 9.2                    |
| CREATION_DATE       | TIMESTAMP    | Not Null   | The date and time the row was created.   | 9.29.2                 |
| MODIFIED_DATE       | TIMESTAMP    | Not Null   | The date and time the row was modified. This column's data is maintained using a database trigger. | 9.2                    |

### 10.5.1.3 PF\_COMMUNITY\_MEMBERSHIP Database Table

The PF\_COMMUNITY\_MEMBERSHIP table contains information regarding the membership status of community members.

**Table 10–83 PF\_COMMUNITY\_MEMBERSHIP Database Table**

| Column Name             | Data Type | Null Value | Description   | New or Changed Version |
|-------------------------|-----------|------------|---|------------------------|
| COMMUNITY_MEMBERSHIP_ID | NUMBER    | Not Null   | PK – A unique, system-generated number to use as the community membership ID. | 9.2                    |
| COMMUNITY_DEFINITION_ID | INTEGER   | Not Null   | FK - A foreign key to PF_COMMUNITY_DEFINITION.                                | 9.2                    |
| COMMUNITY_MEMBER_ID     | INTEGER   | Not Null   | FK - A foreign key to PF_COMMUNITY_MEMBER.                                    | 9.2                    |
| MEMBERSHIP_STATUS       | NUMBER    | Not Null   | Setting to determine membership status. Values are: 1 = active, 0 = Disabled  | 9.2                    |

**Table 10–83 (Cont.) PF\_COMMUNITY\_MEMBERSHIP Database Table**

| Column Name      | Data Type | Null Value | Description  | New or Changed Version |
|------------------|-----------|------------|--|------------------------|
| CREATION_DATE    | TIMESTAMP | Not Null   | The date and time the row was created.   | 9.2                    |
| MODIFIED_DATE    | TIMESTAMP | Not Null   | The date and time the row was modified. This column's data is maintained using a database trigger.   | 9.2                    |
| LAST_ACCESS_DATE | TIMESTAMP | Null       | If access tracking is enabled, this field contains a timestamp that is updated at each login event by the user associated with this CommunityMembership to the linked Community desktop. | 9.2                    |

#### 10.5.1.4 PF\_MEMBERSHIP\_CAPABILITY Database Table

The PF\_MEMBERSHIP\_CAPABILITY table contains information about the capabilities of a member within the community.

**Table 10–84 PF\_COMMUNITY\_MEMBERSHIP Database Table**

| Column Name             | Data Type   | Null Value | Description   | New or Changed Version |
|-------------------------|-------------|------------|---|------------------------|
| COMMUNITY_MEMBERSHIP_ID | INTEGER     | Not Null   | PK/FK to COMMUNITY_MEMBERSHIP.  | 9.2                    |
| CAPABILITY              | VARCHAR(80) | Not Null   | PK - Text label applied to the community member. The only default security functionality with capabilities is that a community "creator" can delete a community and a community "owner" can manage a community. | 9.2                    |

#### 10.5.1.5 PF\_COMMUNITY\_PROPERTY Database Table

The PF\_COMMUNITY\_PROPERTY table contains information regarding custom community properties, used to uniquely identify the community and its characteristics.

**Table 10–85 PF\_COMMUNITY\_PROPERTY Database Table**

| Column Name             | Data Type    | Null Value | Description  | New or Changed Version |
|-------------------------|--------------|------------|--|------------------------|
| COMMUNITY_DEFINITION_ID | INTEGER      | Not Null   | PK/FK – A foreign key to PF_COMMUNITY_DEFINITION.  | 9.2                    |
| PROPERTY_NAME           | VARCHAR(100) | Not Null   | PK – The property name.  | 9.2                    |
| CREATION_DATE           | TIMESTAMP    | Not Null   | The date and time the row was created.   | 9.2                    |
| MODIFIED_DATE           | TIMESTAMP    | Not Null   | The date and time the row was modified. This column's data is maintained using a database trigger. | 9.2                    |
| IS_MULTI_VALUED         | NUMBER       | Not Null   | Boolean value.   | 9.2                    |
| DESCRIPTION             | VARCHAR(255) | Null       | Descriptive text to for this property.   | 9.2                    |

#### 10.5.1.6 PF\_COMMUNITY\_PROPERTY\_VALUE Database Table

The PF\_COMMUNITY\_PROPERTY\_VALUE table contains information regarding the actual value of the custom community property.

**Table 10–86 PF\_COMMUNITY\_PROPERTY\_VALUE Database Table**

| Column Name                 | Data Type    | Null Value | Description  | New or Changed Version |
|-----------------------------|--------------|------------|--|------------------------|
| COMMUNITY_PROPERTY_VALUE_ID | INTEGER      | Not Null   | PK – A system-generated unique number to use as the community property value ID.                   | 9.2                    |
| COMMUNITY_DEFINITION_ID     | INTEGER      | Not Null   | A foreign key to PF_COMMUNITY_PROPERTY.  | 9.2                    |
| PROPERTY_NAME               | VARCHAR(100) | Not Null   | A foreign key to PF_COMMUNITY_PROPERTY.  | 9.2                    |
| CREATION_DATE               | TIMESTAMP    | Not Null   | The date and time the row was created.   | 9.2                    |
| MODIFIED_DATE               | TIMESTAMP    | Not Null   | The date and time the row was modified. This column's data is maintained using a database trigger. | 9.2                    |
| PROPERTY_VALUE              | VARCHAR(400) | Null       | The value associated with the property name.   | 9.2                    |

#### 10.5.1.7 PF\_FILE\_TEMPLATE Database Table

The PF\_FILE\_TEMPLATE table records the location of a file type template. This includes the web application and the URI within the web application to the file, as well as an intersection ID that can be used to retrieve a localized name for the file template.

**Table 10–87 PF\_FILE\_TEMPLATE Database Table**

| Column Name     | Data Type    | Null Value | Description   | New or Changed Version |
|-----------------|--------------|------------|---|------------------------|
| TEMPLATE_ID     | INTEGER      | Not Null   | PK – A system-generated unique identifier for the file template.                            | 9.2                    |
| INTERSECTION_ID | INTEGER      | Not Null   | A foreign key to L10N_RESOURCE for the Localization Resource.                               | 9.2                    |
| TEMPLATE_URI    | VARCHAR(255) | Not Null   | URI pointing to the .portal file that this FileTemplate is based on.                        | 9.2                    |
| WEBAPP_NAME     | VARCHAR(80)  | Not Null   | Name of the web application that contains the .portal file that this FileTemplate is based. | 9.2                    |

#### 10.5.1.8 PF\_INVITATION Database Table

The PF\_INVITATION table stores data related to community invitations.

**Table 10–88 PF\_INVITATION Database Table**

| Column Name              | Data Type | Null Value | Description  | New or Changed Version |
|--------------------------|-----------|------------|--|------------------------|
| INVITATION_ID            | INTEGER   | Not Null   | PK – A system-generated unique identifier for the invitation ID.   | 9.2                    |
| COMMUNITY_DEFINITION_ID  | INTEGER   | Not Null   | A foreign key to PF_COMMUNITY_DEFINITION.  | 9.2                    |
| EXPIRATION_DATE          | TIMESTAMP | Null       | The expiration date for the Invitation, after which the Invitation is no longer active and cannot be consumed.   | 9.2                    |
| IS_SELF_DESTRUCT_ENABLED | NUMBER    | Not Null   | Indicates whether the framework should automatically clean up database information about the invitation after it has been consumed or timed out, or stored for possible later use in queries. Default = 0 (not enabled/false). | 9.2                    |

**Table 10–88 (Cont.) PF\_INVITATION Database Table**

| Column Name        | Data Type    | Null Value | Description   | New or Changed Version |
|--------------------|--------------|------------|---|------------------------|
| CREATION_DATE      | DATE         | Not Null   | The date and time the row was created.  |                        |
| INVITATION_PAYLOAD | VARCHAR(200) | Null       | Application-specific text data that can be attached to an invitation, for use by a communities application during member registration based on this Invitation. | 9.2                    |
| FROM_WLS_USER_NAME | VARCHAR(200) | Null       | The WebLogic Server user name from which the invitation originated  | 9.2                    |

### 10.5.1.9 PF\_INVITEE Database Table

The PF\_INVITEE table contains information regarding the target of the invitation.

**Table 10–89 PF\_INVITEE Database Table**

| Column Name       | Data Type    | Null Value | Description  | New or Changed Version |
|-------------------|--------------|------------|--|------------------------|
| INVITEE_ID        | INTEGER      | Not Null   | PK – A system-generated unique identifier for the invitee ID.  | 9.2                    |
| INVITATION_ID     | INTEGER      | Not Null   | A foreign key to PF_INVITATION.  | 9.2                    |
| INVITEE_ADDRESS   | VARCHAR(255) | Null       | The messaging address (usually an e-mail address) used to send invitations to external users.                                      | 9.2                    |
| INVITATION_STATUS | NUMBER       | Not Null   | The status of the invitation. Values are:<br>10 - Sent (default)<br>20 - Accepted<br>30 - Rejected<br>40 - Revoked<br>50 - Expired | 9.2                    |

**Table 10–89 (Cont.) PF\_INVITEE Database Table**

| Column Name                | Data Type    | Null Value | Description   | New or Changed Version |
|----------------------------|--------------|------------|---|------------------------|
| MODIFIED_DATE              | TIMESTAMP    | Not Null   | Date the status was updated.  | 9.2                    |
| USER_NAME                  | VARCHAR(200) | Null       | The WebLogic Server user name if the invitation is for an existing user, but can be null when inviting external invitees who do not yet have a WebLogic Server user name.                       | 9.2                    |
| INVITATION_VALIDATION_TEXT | VARCHAR(400) | Null       | Validation text that is sent as part of the notification, which must then be provided by the invitee when accepting or declining an invitation. The format of this text is developer-definable. | 9.2                    |

**10.5.1.10 PF\_INVITEE\_PROPERTY Database Table**

The PF\_INVITEE\_PROPERTY table contains information regarding persistent invitee properties such as invitee attributes and capabilities

**Table 10–90 PF\_INVITEE\_PROPERTY Database Table**

| Column Name    | Data Type    | Null Value | Description  | New or Changed Version |
|----------------|--------------|------------|--|------------------------|
| INVITEE_ID     | INTEGER      | Not Null   | PK/FK - A foreign key to PF_INVITEE.   | 9.2                    |
| PROPERTY_TYPE  | NUMBER       | Not Null   | PK - Property type.<br>Possible values are:<br>10 - Persisted Attribute<br>20 - Persisted Capability | 9.2                    |
| PROPERTY_NAME  | VARCHAR(100) | Not Null   | PK - Property name.  | 9.2                    |
| PROPERTY_VALUE | VARCHAR(400) | Not Null   | Value of the persisted attribute or capability.  | 9.2                    |

**10.5.1.11 PF\_MESSAGING\_ADDR Database Table**

The PF\_MESSAGING\_ADDR table is reserved for future use.

**10.5.1.12 PF\_NOTIFICATION Database Table**

The PF\_NOTIFICATION table stores information about event notifications. Notifications for the Portal Framework are a mechanism for publishing availability, and subscribing to and delivering notification events within a portal. A notification event is a sibling to other types of portlet events as defined in the current inter-portlet eventing design, in that portlets can be authored in a declarative manner to listen for notifications using an event subscription tag within the portlet or portlet instance tag. The primary difference between notification events and normal portlet events is that

the source of the notification comes from outside the user's portal desktop. Instead, notifications can come from other users, web applications, or even the framework itself.

**Table 10–91 PF\_NOTIFICATION Database Table**

| Column Name     | Data Type    | Null Value | Description  | New or Changed Version |
|-----------------|--------------|------------|--|------------------------|
| NOTIFICATION_ID | INTEGER      | Not Null   | PK – A system-generated unique identifier for the notification ID.   | 9.2                    |
| USER_NAME       | VARCHAR(200) | Not Null   | User name of the notification recipient.   | 9.2                    |
| PAYLOAD_ID      | INTEGER      | Not Null   | A foreign key to PF_NOTIFICATION_PAYLOAD.  | 9.2                    |
| IS_CONSUMED     | NUMBER       | Not Null   | Indicates whether the notification has been consumed; the notification remains active until it is explicitly consumed or until the expiration time occurs. Default = 0 (not consumed). | 9.2                    |

#### 10.5.1.13 PF\_NOTIFICATION\_PAYLOAD Database Table

The PF\_NOTIFICATION\_PAYLOAD table stores information about notifications for individual users. Because a notification can be sent to groups of users and each user must individually acknowledge (remove) the notification, the notification table contains an entry for each user for each notification. Because the *payload* and other characteristics of a single notification batch are identical for each user, this information is separated and stored only once per notification to minimize redundancy in the database.

For portal desktops that are enabled for notification delivery, WebLogic Portal performs a query against the active notifications for a user prior to executing the portal lifecycle for each request.



**Table 10–92 PF\_NOTIFICATION\_PAYLOAD Database Table**

| Column Name              | Data Type   | Null Value | Description   | New or Changed Version |
|--------------------------|-------------|------------|---|------------------------|
| PAYLOAD_ID               | INTEGER     | Not Null   | PK – A system-generated unique identifier for the payload ID.   | 9.2                    |
| NOTIFICATION_NAMESPACE   | VARCHAR(80) | Not Null   | Denotes the namespace for this Notification, which is should be generated as a unique identifier by the application that constructed the Notification. This is meant to help in avoiding collisions due to duplicate NOTIFICATION_NAMES used by applications. For example, this could be the company name of the application developer that makes use of the Notifications framework. | 9.2                    |
| NOTIFICATION_NAME        | VARCHAR(40) | Not Null   | Name of the notification.   | 9.2                    |
| IS_SELF_DESTRUCT_ENABLED | NUMBER      | Not Null   | Indicates whether the notification framework should automatically clean up database information about the notification after it has been consumed or timed out, or stored for possible later use in queries.<br><br>Default = 0 (not enabled/false).  | 9.2                    |
| IS_ACTIVE                | NUMBER      | Not Null   | Indicates whether the notification is active. When a notification is first sent, its status is set to active for each user. Default = 1 (active).   | 9.2                    |
| EXPIRATION_DATE          | TIMESTAMP   | Not Null   | Expiration time for the notification; the notification remains active until it is explicitly consumed or until the expiration time occurs.  | 9.2                    |
| SOURCE_WEBAPP_NAME       | VARCHAR(80) | Not Null   | Name of the source web application.   | 9.2                    |
| PORTAL_PATH              | VARCHAR(40) | Null       | Path for the portal.  | 9.2                    |
| DESKTOP_PATH             | VARCHAR(40) | Null       | Path for the desktop.   | 9.2                    |

**Table 10–92 (Cont.) PF\_NOTIFICATION\_PAYLOAD Database Table**

| Column Name              | Data Type     | Null Value | Description                         | New or Changed Version |
|--------------------------|---------------|------------|-------------------------------------|------------------------|
| DESKTOP_INSTANCE_TITLE   | VARCHAR(20)   | Null       | Title of the desktop instance.      | 9.2                    |
| PORTLET_DEFINITION_LABEL | VARCHAR(80)   | Null       | Definition label of the portlet.    | 9.2                    |
| PORTLET_INSTANCE_LABEL   | VARCHAR(80)   | Null       | Instance label of the portlet.      | 9.2                    |
| TARGET_WEBAPP_NAME       | VARCHAR(80)   | Null       | Name of the target web application. | 9.2                    |
| PAYLOAD                  | VARCHAR(4000) | Null       | The content of the notification.    | 9.2                    |

## 10.6 Content Management Database Objects

WebLogic Portal's content management system allows you to store content, track its progress, and incorporate content in your portal applications. It provides an easy integration between creating content and delivering that content to your users. Content creators can use WebLogic Portal's repositories to create content and portal developers use the content API and JSP tools to deliver content to portal visitors.

For an entity relation diagram of the Content Management database tables, see [Figure B–11, "Content Management Database Tables"](#).

### 10.6.1 Content Management Data Dictionary Tables

The Content Management system has the following tables:

- [Section 10.6.1.1, "CM\\_NODE Database Table"](#)
- [Section 10.6.1.2, "CM\\_COLLABORATION\\_API\\_METADATA Database Table"](#)
- [Section 10.6.1.4, "CM\\_PROPERTY Database Table"](#)
- [Section 10.6.1.5, "CM\\_PROPERTY\\_CHOICE Database Table"](#)
- [Section 10.6.1.6, "CM\\_PROPERTY\\_DEFINITION Database Table"](#)
- [Section 10.6.1.7, "CM\\_PROPERTY\\_TEXT Database Table"](#)
- [Section 10.6.1.8, "CM\\_WORKFLOW Database Table"](#)

#### 10.6.1.1 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 10–93 CM\_NODE Table Metadata**

| <b>Column Name</b> | <b>Data Type</b> | <b>Null Value</b> | <b>Description</b>  | <b>New or Changed Version</b> |
|--------------------|------------------|-------------------|---|-------------------------------|
| NODE_ID            | INTEGER          | Not Null          | PK – A unique, system-generated number to use as the record ID.   |                               |
| PARENT_NODE_ID     | INTEGER          | Null              | FK – The node's parent record ID (NODE_ID).   |                               |
| CREATION_DATE      | DATETIME         | Not Null          | The date and time the row was created.  |                               |
| MODIFIED_DATE      | DATETIME         | Not Null          | The date and time the row was last modified. This column's data is maintained using a database trigger.                   |                               |
| OBJECT_CLASS_ID    | INTEGER          | Null              | FK – The object class ID associated to the node.  |                               |
| 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          | SMALLINT         | Not Null          | The node type. Possible values are:<br>0 = Unpopulated value<br>1 = Hierarchy Node<br>2 = 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          | ID of the user who created the node.  |                               |
| MODIFIED_BY        | VARCHAR(100)     | Null              | ID of the user who last modified the node.  |                               |
| CM_CREATION_DATE   | DATETIME         | Not Null          | The date and time the row was created. Maintained by the application.   |                               |
| CM_MODIFIED_DATE   | DATETIME         | Not Null          | The date and time the row was last modified. Maintained by the application.   |                               |

**Table 10–93 (Cont.) CM\_NODE Table Metadata**

| Column Name         | Data Type    | Null Value | Description   | New or Changed Version |
|---------------------|--------------|------------|---|------------------------|
| FULL_PATH           | VARCHAR(882) | Null       | AK – Each node has a path that uniquely identifies it within the repository.<br><br>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.<br><br>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/. | 9.2                    |
| LIFECYCLE_STATUS    | INTEGER      | Null       | The specific workflow status that the node version has been assigned:   |                        |
| WORKFLOW_ID         | INTEGER      | Null       | FK – The Workflow ID associated with the node.  |                        |
| CHILD_NODE_POSITION | NUMERIC(13)  | Not Null   | The position of the child node with respect to other child nodes in the hierarchy. New children are added at the end of the grouping. Nodes with no current position are assigned a default value of -1.  | 10.0                   |

### 10.6.1.2 CM\_COLLABORATION\_API\_METADATA Database Table

This table is used to store information about WebLogic Portal's collaboration portlets.

**Table 10–94 CM\_COLLABORATION\_API\_METADATA Table Metadata**

| Column Name                    | Data Type | Null Value | Description   | New or Changed Version |
|--------------------------------|-----------|------------|---|------------------------|
| NODE_ID                        | INTEGER   | Not Null   | PK - A unique, system-generated number to use as the record ID.                   | 9.2                    |
| ICONTAINER_SUBCONTAINER_COUNT  | INTEGER   | Not Null   | The number of shallow child containers for the current node.                      | 9.2                    |
| IGWARE_CONTR_TOTAL_ITEM_COUNT  | INTEGER   | Not Null   | The number of shallow child items for the current node.                           | 9.2                    |
| IGWARE_CONTR_UNREAD_ITEM_COUNT | INTEGER   | Not Null   | The number of shallow child items that are marked as UNREAD for the current node. | 9.2                    |

**Table 10–94 (Cont.) CM\_COLLABORATION\_API\_METADATA Table Metadata**

| Column Name                    | Data Type | Null Value | Description   | New or Changed Version |
|--------------------------------|-----------|------------|---|------------------------|
| FORUM_CATEGORY_FORUM_COUNT     | INTEGER   | Not Null   | The number of shallow forums beneath this category.     | 9.2                    |
| ITOPIC_SUPPRT_DEEP_TOPIC_COUNT | INTEGER   | Not Null   | The number of topics, infinitely deep, below this node. | 9.2                    |
| ITOPIC_SUPPRT_TOPIC_COUNT      | INTEGER   | Not Null   | The number of shallow topics below this node.           | 9.2                    |

### 10.6.1.3 CM\_OBJECT\_CLASS Database Table

The ObjectClass is the schema for a Node (a content type). 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 10–95 CM\_OBJECT\_CLASS Table Metadata**

| Column Name                    | Data Type     | Null Value | Description  | New or Changed Version |
|--------------------------------|---------------|------------|--|------------------------|
| OBJECT_CLASS_ID                | INTEGER       | Not Null   | PK – A unique, system-generated number to use as the record ID.  |                        |
| CREATION_DATE                  | DATETIME      | Not Null   | The date and time the row was created.   |                        |
| MODIFIED_DATE                  | DATETIME      | 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 | INTEGER       | 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. |                        |

**Table 10–95 (Cont.) CM\_OBJECT\_CLASS Table Metadata**

| Column Name              | Data Type     | Null Value | Description   | New or Changed Version |
|--------------------------|---------------|------------|---|------------------------|
| IS_ABSTRACT              | SMALLINT      | Not Null   | A flag indicating if this type (object class) is abstract. An abstract type cannot be used as the object class of a node, but can be used as a base type for others using type inheritance or as a nested type. Possible values are:<br>0 = false<br>1 = true | 9.2                    |
| IS_SEARCHABLE            | SMALLINT      | Not Null   | A flag indicating if this property is searchable.   | 9.2                    |
| PARENT_OBJECT_CLASS_ID   | INTEGER       | Null       | FK – The OBJECT_CLASS_ID of the parent type if this type is inheriting from a parent type.  | 9.2                    |
| PATH                     | VARCHAR (882) | Not Null   | The string path of the object class. This will take the form of /object_class_name for most object classes. Those that use inheritance will take the form /parent/child or /grandparent/parent/child, and so on.  | 9.2                    |
| OBJECT_CLASS_DESCRIPTION | VARCHAR (254) | Null       | Optional description for the content type.  | 9.2                    |
| WORKFLOW_ID              | INTEGER       | Null       | The workflow ID associated with the type.   | 9.2                    |

#### 10.6.1.4 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 10–96 CM\_PROPERTY Table Metadata**

| Column Name   | Data Type | Null Value | Description   | New or Changed Version |
|---------------|-----------|------------|---|------------------------|
| PROPERTY_ID   | INTEGER   | Not Null   | PK – A unique, system-generated number to use as the record ID.   |                        |
| CREATION_DATE | DATETIME  | Not Null   | The date and time the row was created.  |                        |
| MODIFIED_DATE | DATETIME  | Not Null   | The date and time the row was last modified. This column's data is maintained using a database trigger. |                        |
| NODE_ID       | INTEGER   | Not Null   | FK – The ID of the node that contains the property.   |                        |

**Table 10–96 (Cont.) CM\_PROPERTY Table Metadata**

| <b>Column Name</b>      | <b>Data Type</b> | <b>Null Value</b> | <b>Description</b>   | <b>New or Changed Version</b> |
|-------------------------|------------------|-------------------|--|-------------------------------|
| PROPERTY_NAME           | VARCHAR(894)     | Not Null          | The name of the property. It must be unique relative to its node.  | 9.2                           |
| PROPERTY_TYPE           | SMALLINT         | Not Null          | The type of the property:<br>0 = Boolean<br>1 = Number<br>2 = Float<br>3 = Varchar<br>4 = Date<br>5 = BLOB<br>6 = Nested<br>7 = Linked |                               |
| PROPERTY_DEFINITION_ID  | INTEGER          | Null              | FK – The ID of the property definition to which this property must conform.  |                               |
| BOOLEAN_VALUE           | SMALLINT         | Null              | True (1) for the Property if the PROPERTY_TYPE is Boolean (PROPERTY_TYPE=0).   |                               |
| DATETIME_VALUE          | DATETIME         | Null              | The datetime value for the property if the PROPERTY_TYPE is DATE (PROPERTY_TYPE=4).  |                               |
| LONG_VALUE              | NUMERIC          | 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_VALUE_CHECKSUM_HEX | VARCHAR(32)      | Null              | The HEX representation of the MD5 checksum for each binary value in the table.   | 9.2                           |
| BLOB_FILE_NAME          | VARCHAR(50)      | Null              | Name of the file associated with the BLOB_VALUE.   |                               |

**Table 10–96 (Cont.) CM\_PROPERTY Table Metadata**

| Column Name                 | Data Type    | Null Value | Description   | New or Changed Version |
|-----------------------------|--------------|------------|---|------------------------|
| BLOB_FILE_SIZE              | INTEGER      | 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 character set) for the BLOB_VALUE. For example: "text/html;charset=iso8859-1"   |                        |
| LINKED_NODE_REPOSITORY_NAME | VARCHAR(254) | Null       | For a property of type link, the name of the repository to which the link property refers.  | 9.2                    |
| LINKED_NODE_UID             | VARCHAR(254) | Null       | For a property of type link, the repository-specific node UID to which the link property refers.  | 9.2                    |
| NESTED_GROUP_ID             | VARCHAR(254) | Null       | An ID that acts as a grouping mechanism to relate a set of multi-valued nested properties together.   | 9.2                    |
| ORDERED_POSITION            | SMALLINT     | Not Null   | A number associated with multi-valued properties for deterministic ordering in the presentation of the property values. For example, if the property values Snow, Sleet, Wind and Rain are added when constructing a multi-valued property those values will always appear in the order in which they were added. Property value Snow would have an ordered position of 1, Sleet would be 2 and so on. The property values ordered position can be updated (or re-ordered) with the Content Management API. | 10.0                   |

A number associated with multi-valued properties for deterministic ordering in the presentation of the property values. For example, if the property values Snow, Sleet, Wind and Rain are added when constructing a multi-valued property those values will always appear in the order in which they were added. Property value Snow would have an ordered position of 1, Sleet would be 2 and so on. The property values ordered position can be updated (or re-ordered) via the Content Management API.

#### 10.6.1.5 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\_PROPERTY=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 10–97 CM\_PROPERTY\_CHOICE Table Metadata**

| Column Name             | Data Type    | Null Value | Description   | New or Changed Version |
|-------------------------|--------------|------------|---|------------------------|
| PROPERTY_CHOICE_ID      | INTEGER      | Not Null   | PK – A unique, system-generated number to use as the record ID.   |                        |
| CREATION_DATE           | DATETIME     | Not Null   | The date and time the row was created.  |                        |
| MODIFIED_DATE           | DATETIME     | Not Null   | The date and time the row was last modified. This column's data is maintained using a database trigger.   |                        |
| PROPERTY_DEFINITION_ID  | INTEGER      | Not Null   | FK – The ID of the property definition that contains the property choice.                                 |                        |
| DEFAULT_PROPERTY        | SMALLINT     | Not Null   | Set to 1 if the property choice is a default, or 0 if it is not.  |                        |
| BOOLEAN_VALUE           | SMALLINT     | Null       | True (1) for the Property if the PROPERTY_TYPE is BOOLEAN (PROPERTY_TYPE=0).                              |                        |
| DATETIME_VALUE          | DATETIME     | Null       | The date/time value for the Property if the PROPERTY_TYPE is DATE (PROPERTY_TYPE=4).                      |                        |
| LONG_VALUE              | NUMERIC      | 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_VALUE_CHECKSUM_HEX | VARCHAR (32) | Null       | The HEX representation of the MD5 checksum for each binary value in the table.                            | 9.2                    |

**Table 10–97 (Cont.) CM\_PROPERTY\_CHOICE Table Metadata**

| Column Name       | Data Type    | Null Value | Description  | New or Changed Version |
|-------------------|--------------|------------|--|------------------------|
| BLOB_FILE_NAME    | VARCHAR(50)  | Null       | The name of the file associated with the BLOB_VALUE.   |                        |
| BLOB_FILE_SIZE    | INTEGER      | 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 charset) for the BLOB_VALUE. For example: "text/html;charset=iso8859-1"  |                        |
| ORDERED_POSITION  | SMALLINT     | Not Null   | A number associated with property choices for deterministic ordering in the presentation of the property choices. The default is 0. For example, if the property choices Snow, Sleet, Wind, and Rain are added when constructing the property choices for a given property, those choices will always appear in the order in which they were added. Property choice Snow would have an ordered position of 1, Sleet would be 2 and so on. The property choices ordered positions can be updated or re-ordered with the Content Management API. | 10.2                   |

#### 10.6.1.6 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 other than *green*, *red*, *blue*, or null.

**Table 10–98 CM\_PROPERTY\_DEFINITION Table Metadata**

| Column Name            | Data Type    | Null Value | Description   | New or Changed Version |
|------------------------|--------------|------------|---|------------------------|
| PROPERTY_DEFINITION_ID | INTEGER      | Not Null   | PK – A unique, system-generated number to use as the record ID.   |                        |
| CREATION_DATE          | DATETIME     | Not Null   | The date and time the row was created.  |                        |
| MODIFIED_DATE          | DATETIME     | Not Null   | The date and time the row was last modified. This column's data is maintained using a database trigger.   |                        |
| OBJECT_CLASS_ID        | INTEGER      | 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          | SMALLINT     | Not Null   | The type of the property:<br>0 = Boolean<br>1 = Number<br>2 = Float<br>3 = Varchar<br>4 = Date<br>5 = BLOB  |                        |
| IS_MANDATORY           | SMALLINT     | Not Null   | True if the value of a property must be set.  |                        |
| IS_READ_ONLY           | SMALLINT     | Not Null   | True if the value of a property should not be set by an end user.   |                        |
| IS_RESTRICTED          | SMALLINT     | Not Null   | True if the value of a property should come from the property choice values.  |                        |
| IS_MULTI_VALUED        | SMALLINT     | 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.   |                        |

**Table 10–98 (Cont.) CM\_PROPERTY\_DEFINITION Table Metadata**

| Column Name              | Data Type | Null Value | Description   | New or Changed Version |
|--------------------------|-----------|------------|---|------------------------|
| IS_SEARCHABLE            | SMALLINT  | Not Null   | Defines whether values of this property definition should be indexed by the full text search engine. Possible values are:<br><br>0 = Not searchable (the default)<br><br>1 = Searchable   | 9.2                    |
| PROPERTY_DEFINITION_TYPE | SMALLINT  | Not Null   | The type of the property definition. Native definitions are defined on the containing object class. Inherited definitions are populated through type inheritance, and overridden definitions are also populated through type inheritance, but changed by the inheriting (child) type. Possible values are:<br><br>1 = Native (the default)<br><br>2 = Inherited<br><br>3 = Overridden | 9.2                    |
| NESTED_OBJECT_CLASS_ID   | INTEGER   | Not Null   | FK – The OBJECT_CLASS_ID of the nested type. An object class can "nest" the set of definitions as defined by another object class.  | 9.2                    |

#### 10.6.1.7 CM\_PROPERTY\_TEXT Database Table

This table provides for large textual property value strings. The View CM\_PROPERTY\_V (new in 10.0) joins the CM\_PROPERTY\_TEXT table with the CM\_PROPERTY table to expand the size of text value strings to VARCHAR(4254).

**Table 10–99 CM\_PROPERTY\_TEXT Table Metadata**

| Column Name    | Data Type      | Null Value | Description   | New or Changed Version |
|----------------|----------------|------------|---|------------------------|
| PROPERTY_ID    | INTEGER        | Not Null   | PK and FK – The property ID from the CM_PROPERTY table. | 10.0                   |
| TEXT_VALUE_EXT | VARCHAR (4000) | Null       | Extends textual property value strings.                 | 10.0                   |

#### 10.6.1.8 CM\_WORKFLOW Database Table

This table was added in WebLogic Portal version 9.2. It stores information for the Content Management workflow.

**Table 10–100 CM\_WORKFLOW Table**

| Column Name       | Data Type    | Null Value | Description   | New or Changed Version |
|-------------------|--------------|------------|---|------------------------|
| WORKFLOW_ID       | INTEGER      | Not Null   | PK – A unique, system-generated number to use as the workflow ID.                                       | 9.2                    |
| WORKFLOW_NAME     | VARCHAR (50) | Not Null   | The name of the workflow.   | 9.2                    |
| WORKFLOW_DOCUMENT | BLOB         | Not Null   | The binary representation of the workflow document.   | 9.2                    |
| CREATION_DATE     | DATETIME     | Not Null   | The date and time the row was created.  | 9.2                    |
| MODIFIED_DATE     | DATETIME     | Not Null   | The date and time the row was last modified. This column's data is maintained using a database trigger. | 9.2                    |
| WORKFLOW_COMMENT  | VARCHAR(254) | Null       | Optional comment for the workflow.  | 9.2                    |

## 10.7 Content Management Virtual Content Repository Database Objects

The WLP repository provides automatic versioning.

For an entity relation diagram of the Virtual Content Repository (versioning) database tables see [Figure B–12, "Content Management Virtual Content Repository Database Tables"](#).

### 10.7.1 Content Management Virtual Object Data Dictionary Tables

The Content Management system has the following tables for virtual objects:

- [Section 10.7.1.1, "CMV\\_NODE Table"](#)
- [Section 10.7.1.2, "CMV\\_NODE\\_ASSIGNED\\_ROLE Table"](#)
- [Section 10.7.1.3, "CMV\\_NODE\\_VERSION Table"](#)
- [Section 10.7.1.4, "CMV\\_PROPERTY Table"](#)
- [Section 10.7.1.5, "CMV\\_VALUE Table"](#)
- [Section 10.7.1.6, "CMV\\_VALUE\\_TEXT Table"](#)
- [Section 10.7.1.7, "CMV\\_NODE\\_VERSION\\_PROPERTY Table"](#)

#### 10.7.1.1 CMV\_NODE Table

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

**Table 10–101 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.           |
| NODE_NAME                  | VARCHAR (50)  | Not Null   | The name of the node.   |
| 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.                                |
| OBJECT_CLASS_ID            | VARCHAR (254) | Null       | The object class ID that is associated with the node.                     |
| ASSIGNED_TO_USER_NAME      | VARCHAR(200)  | Null       | User name to which the node is assigned.                                  |
| LAST_ASSIGNED_TO_USER_NAME | VARCHAR(200)  | Null       | User name to which the node is assigned; to be used in a revert scenario. |

**10.7.1.2 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 10–102 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.   |

**10.7.1.3 CMV\_NODE\_VERSION Table**

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

**Table 10–103 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 | DATETIME     | Not Null   | Time the node version was last edited.   |
| MODIFIED_BY      | VARCHAR(100) | Not Null   | User name 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 life cycle status that the node version has been assigned (for example, In Progress, Published, and so on). |

**10.7.1.4 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 10–104 CMV\_PROPERTY Table Metadata**

| Column Name   | Data Type    | Null Value | Description   | New or Changed Version |
|---------------|--------------|------------|---|------------------------|
| PROPERTY_ID   | VARCHAR(254) | Not Null   | PK – A unique, system-generated number to use as the record ID.   |                        |
| CREATION_DATE | DATETIME     | Not Null   | The date and time the row was created; the default is the current timestamp.  |                        |
| MODIFIED_DATE | DATETIME     | Not Null   | Date and time the row was last modified; the default is the current timestamp. This column's data is maintained using a database trigger. |                        |
| PROPERTY_NAME | VARCHAR(894) | Not Null   | The name of the property. It must be unique relative to its node.   | 9.2                    |
| PROPERTY_TYPE | SMALLINT     | Not Null   | The type of the property:<br>0 = Boolean<br>1 = Number<br>2 = Float<br>3 = Varchar<br>4 = Date<br>5 = BLOB                                |                        |

**10.7.1.5 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 10–105 CMV\_VALUE Table**

| Column Name   | Data Type    | Null Value | Description   | New or Changed Version |
|---------------|--------------|------------|---|------------------------|
| 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.  |                        |
| CREATION_DATE | DATETIME     | Not Null   | The date and time the row was created; the default is the current timestamp. This column's data is maintained using a database trigger. |                        |

**Table 10–105 (Cont.) CMV\_VALUE Table**

| Column Name                 | Data Type     | Null Value | Description   | New or Changed Version |
|-----------------------------|---------------|------------|---|------------------------|
| MODIFIED_DATE               | DATETIME      | Not Null   | Date and time the row was last modified; the default is the current timestamp. 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              | DATETIME      | 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_VALUE_CHECKSUM_HEX     | VARCHAR(32)   | Null       | The HEX representation of the MD5 checksum for each binary value in the table.  | 9.2                    |
| 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"                                  |                        |
| LINKED_NODE_REPOSITORY_NAME | VARCHAR(254)  | Null       | For a property of type link, the name of the repository to which the link property refers.  | 9.2                    |



**Table 10–105 (Cont.) CMV\_VALUE Table**

| Column Name      | Data Type    | Null Value | Description   | New or Changed Version |
|------------------|--------------|------------|---|------------------------|
| LINKED_NODE_UID  | VARCHAR(254) | Null       | For a property of type link, the repository-specific node UID to which the link property refers.  | 9.2                    |
| NESTED_GROUP_ID  | VARCHAR(254) | Null       | An ID that acts as a grouping mechanism to relate a set of multi-valued nested properties together.   | 9.2                    |
| ORDERED_POSITION | SMALLINT     | Not Null   | A number associated with multi-valued properties for deterministic ordering in the presentation of the property values. For example, if the property values Snow, Sleet, Wind and Rain are added when constructing a multi-valued property those values will always appear in the order in which they were added. Property value Snow would have an ordered position of 1, Sleet would be 2 and so on. The property values ordered position can be updated (or re-ordered) with the Content Management API. | 10.0                   |

**10.7.1.6 CMV\_VALUE\_TEXT Table**

This table provides for large textual property value strings. The View CMV\_VALUE\_V (new in 10.0) joins the CMV\_VALUE\_TEXT table with the CMV\_VALUE table to expand the size of text value strings to VARCHAR(4254).

**Table 10–106 CMV\_VALUE\_TEXT Table Metadata**

| Column Name    | Data Type      | Null Value | Description   | New or Changed Version |
|----------------|----------------|------------|---|------------------------|
| PROPERTY_ID    | VARCHAR (254)  | Not Null   | PK and FK – The PROPERTY_ID from the CMV_VALUE table. | 10.0                   |
| VALUE_ID       | VARCHAR (254)  | Not Null   | PK and FK – The value ID from the CMV_VALUE table.    | 10.0                   |
| TEXT_VALUE_EXT | VARCHAR (3746) | Null       | Extends textual value strings.                        | 10.0                   |

**10.7.1.7 CMV\_NODE\_VERSION\_PROPERTY Table**

This table uniquely identifies a relationship between a CMV\_NODE\_VERSION and CMV\_PROPERTY.

**Table 10–107 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(10)  | 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.      |

## 10.8 RDBMS Security Store Tables

WebLogic Server provides the option of using an external RDBMS as a datastore that is used by authorization, role mapping, credential mapping, and certificate registry providers. WebLogic Portal uses this datastore as part of its default domain configuration. When you configure a WLP domain, the RDBMS security store tables are automatically created.

For detailed information on the RDBMS security store, see the WebLogic Server document, "Managing the RDBMS Security Store," in *Oracle Fusion Middleware Securing Oracle WebLogic Server*.

[Table 10–108](#) highlights the collection of columns from the RDBMS security store tables for which WLP has modified the column sizes.

**Table 10–108 RDBMS Security Store Database Table Defaults**

| Column Name | WLS Default Size (bytes) | WLP Default Size (bytes) | DB2         | MySQL |
|-------------|--------------------------|--------------------------|-------------|-------|
| DOMN        | 128                      | 128                      | 50          | 50    |
| REALMN      | 128                      | 128                      | 50          | 50    |
| CREDN       | 400                      | 768                      | WLP default | 600   |
| CN          | 400                      | 768                      | WLP default | 600   |
| PM_CN       | 400                      | 768                      | WLP default | 600   |
| ENTRY_KEY   | 400                      | 768                      | WLP default | 600   |
| WCRIDN      | 400                      | 768                      | WLP default | 600   |
| WCRSN       | 400                      | 768                      | WLP default | 600   |
| WCRSDN      | 400                      | 768                      | WLP default | 600   |
| WCRSKI      | 400                      | 768                      | WLP default | 600   |
| XRS         | 400                      | 768                      | WLP default | 600   |
| TYPEN       | 400                      | 768                      | WLP default | 600   |
| XR          | 400                      | 768                      | WLP default | 600   |

---

---

## Scripts and Properties Files

This appendix describes the SQL scripts and properties files that specify the settings for database creation and upgrade, and perform Database Definition Language (DDL) commands for WebLogic Portal and personalization data structures. It includes the following sections:

- [Section A.1, "WebLogic Portal DDL Modules"](#)
- [Section A.2, "Content Management DDL Modules"](#)
- [Section A.3, "Personalization DDL Modules"](#)
- [Section A.4, "Using the database.properties File"](#)
- [Section A.5, "Scripts to Create an Additional Content Management Repository"](#)
- [Section A.6, "Scripts to Manually Upgrade from Version 8.1, 9.2, 10.0, or 10.2"](#)
- [Section A.7, "Scripts to Drop Deprecated Compoze Database Tables"](#)
- [Section A.8, "Scripts to Drop Deprecated RDBMS Authenticator Tables"](#)

### A.1 WebLogic Portal DDL Modules

WebLogic Portal DDL modules (SQL scripts) are provided in the following directories:

```
<WLPORTAL_HOME>\portal\db\db2
<WLPORTAL_HOME>\portal\db\mysql
<WLPORTAL_HOME>\portal\db\oracle
<WLPORTAL_HOME>\portal\db\pointbase
<WLPORTAL_HOME>\portal\db\sql_server
<WLPORTAL_HOME>\portal\db\sybase
```

Each of these directories, except for `pointbase`, has an `admin` subdirectory, which contains the scripts that create the database user or database, and the appropriate database objects (such as tablespaces, bufferpools, and so on, depending on the database requirements).

Insert commands for bootstrap data that must be inserted into tables in each WebLogic Portal database are provided in the following directory:

```
<WLPORTAL_HOME>\portal\db\data\required\xx_insert_system_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

```

In these file names, `xx` is one of the prefixes listed [Table A-1](#).

**Table A-1** *WebLogic Portal DDL Module File Prefixes*

| Prefix                   | Description                       |
|--------------------------|-----------------------------------|
| au                       | Anonymous user                    |
| comm                     | Communities                       |
| dep9                     | Deprecated Collaboration portlets |
| pf, pf9, pf10, and pf102 | Framework and localization        |
| wps                      | WebLogic Portal Services          |

The scripts with a 10 suffix contain new and updated DDL for WebLogic Portal 10.0. Scripts with a 102 suffix contain new and updated DDL for WebLogic Portal 10.2.

## A.2 Content Management DDL Modules

Content Management DDL modules (SQL scripts) are provided in the following directories:

```

<WLPORTAL_HOME>\content-mgmt\db\db2
<WLPORTAL_HOME>\content-mgmt\db\mysql
<WLPORTAL_HOME>\content-mgmt\db\oracle
<WLPORTAL_HOME>\content-mgmt\db\pointbase
<WLPORTAL_HOME>\content-mgmt\db\sql_server
<WLPORTAL_HOME>\content-mgmt\db\sybase

```

Insert commands for bootstrap data that must be inserted into tables in each content management database are provided in the following directory:

```

<WLPORTAL_HOME>\content-mgmt\db\data\required\xx_insert_system_data.sql

```

Content Management 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_drop_constraints.sql
xx_drop_fkeys.sql
xx_drop_indexes.sql
xx_drop_tables.sql

```

In these file names, `xx` is one of the prefixes listed [Table A-2](#).

**Table A-2** *Content Management DDL Module File Prefixes*

| Prefix                   | Description        |
|--------------------------|--------------------|
| cm, cm9, cm10, and cm102 | Content management |

**Table A–2 (Cont.) Content Management DDL Module File Prefixes**

| Prefix       | Description        |
|--------------|--------------------|
| cmv and cmv9 | Content management |

**Tip:** The scripts with a 10 suffix contain new and updated DDL to Content Management for WebLogic Portal 10.0. Scripts with a 102 suffix contain new and updated DDL to Content Management for WebLogic Portal 10.2.

### A.3 Personalization DDL Modules

Personalization DDL modules (SQL scripts) are provided in the following directories:

```
<WLPORTAL_HOME>\p13n\db\db2
<WLPORTAL_HOME>\p13n\db\mysql
<WLPORTAL_HOME>\p13n\db\oracle
<WLPORTAL_HOME>\p13n\db\pointbase
<WLPORTAL_HOME>\p13n\db\sql_server
<WLPORTAL_HOME>\p13n\db\sybase
```

Insert commands for bootstrap data that must be inserted into tables in each personalization database are provided in the following directory:

```
<WLPORTAL_HOME>\p13n\db\data\required\xx_insert_system_data.sql
```

Personalization 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_drop_constraints.sql
xx_drop_fkeys.sql
xx_drop_indexes.sql
xx_drop_tables.sql
```

In these file names, xx is one of the prefixes listed [Table A–3](#).

**Table A–3 Personalization DDL Module File Prefixes**

| Prefix                   | Description                                       |
|--------------------------|---|
| bt                       | Behavior tracking                                 |
| dep9                     | Deprecated WebLogic Portal RDBMS Authenticator    |
| er                       | Visitor entitlements and delegated administration |
| p13n, p13n9, and p13n102 | Users, groups, and user profiles                  |
| seq                      | Sequencer   |

**Tip:** There are no DDL changes to personalization for WebLogic Portal 10.0. Scripts with a 102 suffix contain new and updated DDL to Personalization for WebLogic Portal 10.2.

### A.4 Using the database.properties File

Database scripts use the database.properties file for the following purposes:

- To connect to the database
- To drop, create, or alter database objects
- To perform data inserts

You can find the `database.properties` file in any domain directory that contains WebLogic Portal. Database scripts, such as `create_db.cmd` and `create_db.sh`, use the `database.properties` file that is located in the same directory from which you start the script.

#### A.4.1 Setting the Database Parameters in the Properties File

Use the `database.properties` file to specify the database you plan to use with WebLogic Portal.

The following is the section in the file where you choose your database vendor.

```
# Set database= to a valid database.
# Valid databases are: pointbase, oracle, sql_server, sybase and db2
#-----
database=pointbase
```

In addition to modifying the database name, you must specify the appropriate values for the database user, password, host, and so on. You can use an encrypted password, as described in [Section 3.1, "Encrypting Passwords."](#) The following shows the PointBase and Oracle sections of the properties file.

```
#-----
# For the database specified above, fill in the appropriate @DB_USER@
# and @DB_PASSWORD@ settings and complete the url parameters by
# setting: @DB_HOST@, @DB_PORT@, @DB_NAME@
# Note: @DB_NAME@ for sql_server and sybase is often the same as user
#
# To use an encrypted password for any database use:
#   weblogic.security.Encrypt to obtain the encrypted password.
# The saltFile used to encrypt/decrypt passwords is the domains
#   <DOMAIN_HOME>/security/SerializedSystemIni.dat saltFile.
#
#-----
pointbase.user=WEBLOGIC
pointbase.password=WEBLOGIC
pointbase.driver=com.pointbase.jdbc.jdbcUniversalDriver
pointbase.url=jdbc:pointbase:server://localhost:9093/weblogic_eval
#-----
oracle.user=@DB_USER@
oracle.password=@DB_PASSWORD@
oracle.driver=weblogic.jdbc.oracle.OracleDriver
oracle.url=jdbc:bea:oracle://@DB_HOST@:@DB_PORT@;SID=@DB_NAME@
#-----
You can specify a log file other than the default.
#-----
# logFile= the file that output will be logged to
#-----
logFile=create_db.log
The files= setting indicates which SQL scripts to execute. The default setting is
appropriate for creating the main WebLogic Portal database.
#-----
# files= points to the jdbc.index files, which points to the .sql
#   files, to be executed by create_db
#-----
files=${env.WEBLOGIC_HOME}/common/p13n/db/${database}/jdbc_index/ jdbc.index,
```

```
$(env.WLP_HOME)/db/${database}/jdbc_index/jdbc.index
```

You can also change the debug setting or send all SQL to a file rather than executing it, as described in the comments in the properties file. You can then execute the SQL file against the database.

## A.5 Scripts to Create an Additional Content Management Repository

If you need to create an additional content management repository, use `create_db.cmd/.sh` command with the `cmrepo_database.properties` file. For more information, see the *Oracle Fusion Middleware Content Management Guide for Oracle WebLogic Portal*.

## A.6 Scripts to Manually Upgrade from Version 8.1, 9.2, 10.0, or 10.2

If you did not use the WebLogic Upgrade Wizard to perform the database upgrade from 8.1, you must upgrade manually. See the *Oracle Fusion Middleware Upgrade Guide for Oracle WebLogic Portal* for details on the upgrade process.

## A.7 Scripts to Drop Deprecated Compoze Database Tables

After you have upgraded to version 10.0, you can drop the tables associated with Compoze (Collaboration). See the *Oracle Fusion Middleware Upgrade Guide for Oracle WebLogic Portal* for instructions.

## A.8 Scripts to Drop Deprecated RDBMS Authenticator Tables

If you do not upgrade your user store using the WebLogic Upgrade Wizard during the domain upgrade process, you can perform a manual upgrade later.

After you have upgraded to the WebLogic SQL Authenticator, you can drop the tables associated with the WebLogic Portal RDBMS Authenticator.

See the *Oracle Fusion Middleware Upgrade Guide for Oracle WebLogic Portal* for instructions.





---

---

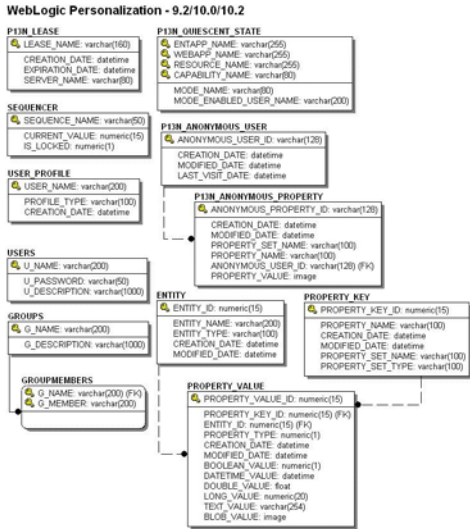
## Database Schema Diagrams

This appendix includes entity relation diagrams for WLP database tables.

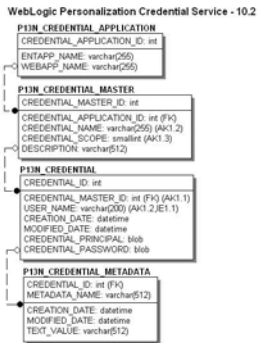
The diagrams include:

- Figure B-1, "Personalization Database Tables"
- Figure B-2, "Personalization Credential Service Database Tables"
- Figure B-3, "Personalization Job Scheduler Database Tables"
- Figure B-4, "Entitlement Database Tables"
- Figure B-5, "Behavior Tracking Database Tables"
- Figure B-6, "Data Synchronization Database Tables"
- Figure B-7, "Portal Services Database Tables"
- Figure B-8, "Portal Framework Database Tables"
- Figure B-9, "WSRP Portal Framework Tables"
- Figure B-10, "Communities Portal Framework Database Tables"
- Figure B-11, "Content Management Database Tables"
- Figure B-12, "Content Management Virtual Content Repository Database Tables"

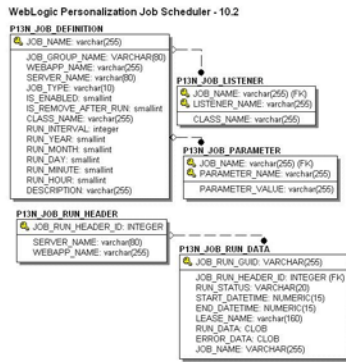
**Figure B-1 Personalization Database Tables**



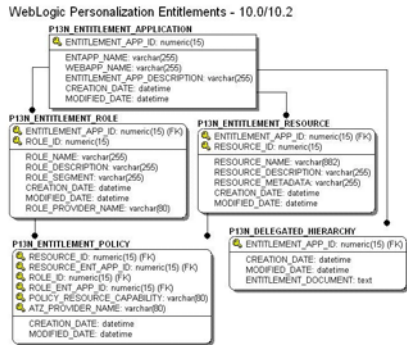
**Figure B-2 Personalization Credential Service Database Tables**



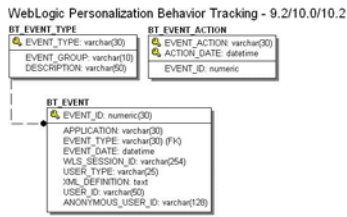
**Figure B-3 Personalization Job Scheduler Database Tables**



**Figure B-4 Entitlement Database Tables**



**Figure B-5 Behavior Tracking Database Tables**



**Figure B-6 Data Synchronization Database Tables**

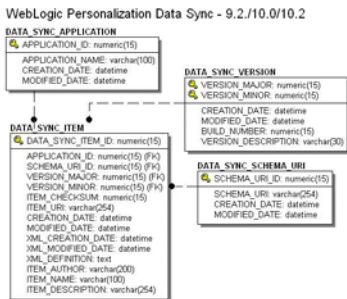




Figure B-9 WSRP Portal Framework Tables

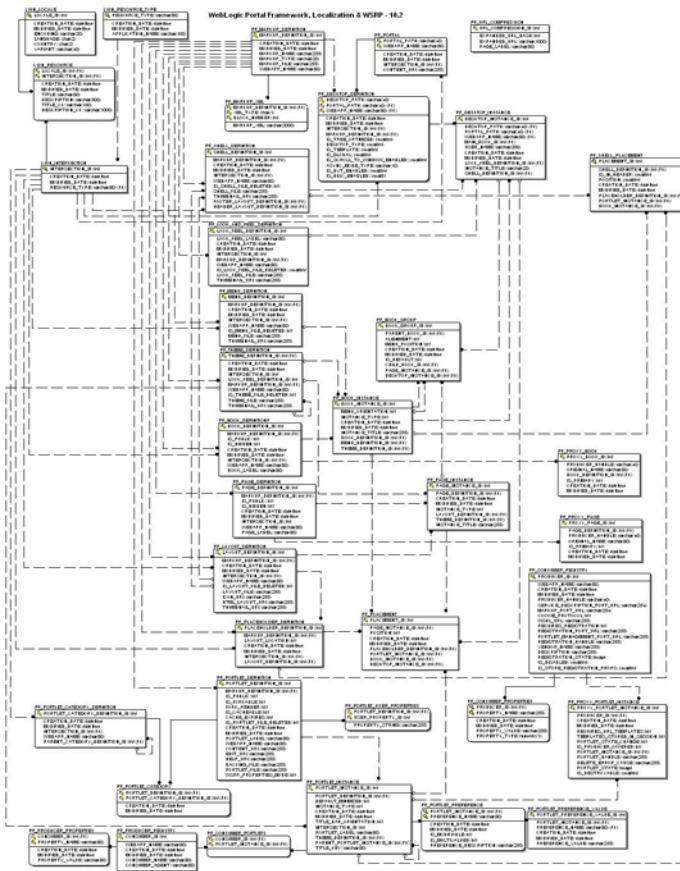
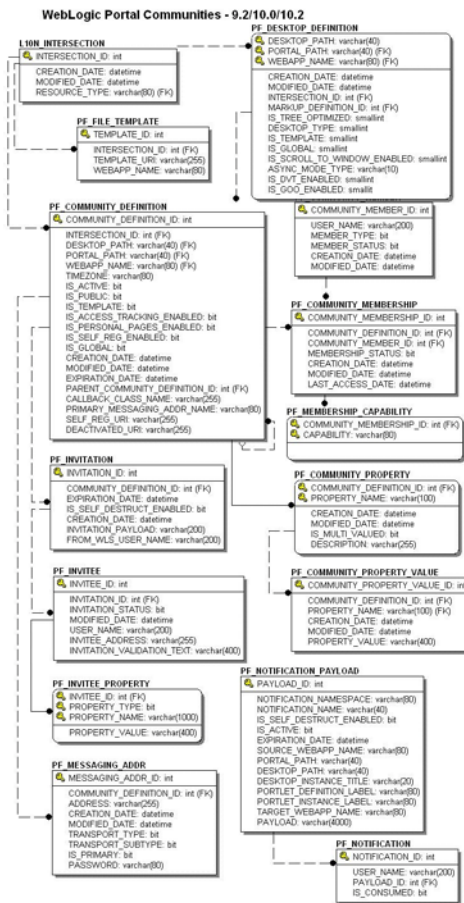
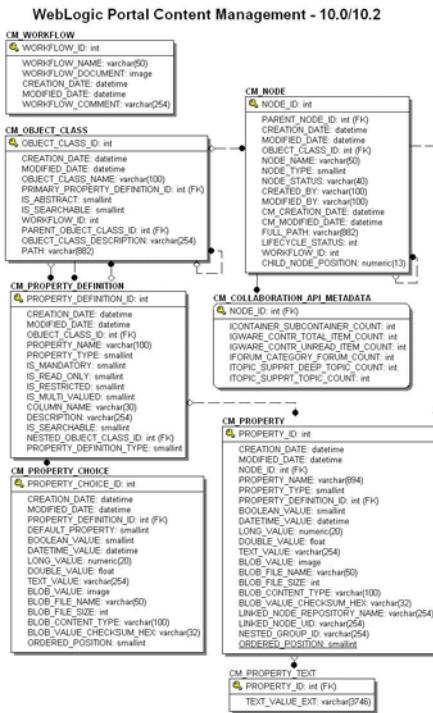


Figure B-10 Communities Portal Framework Database Tables



**Figure B-11 Content Management Database Tables**



**Figure B-12 Content Management Virtual Content Repository Database Tables**

