

Oracle9iAS Reports Services

レポート Web 公開ガイド

リリース 9.0.2

2002 年 8 月

部品番号 : J05993-01

ORACLE®

Oracle9iAS Reports Services レポート Web 公開ガイド, リリース 9.0.2

部品番号 : J05993-01

原本名 : Oracle9iAS Reports Services Publishing Reports to the Web, Release 2 (9.0.2)

原本部品番号 : A92102-01

原著者 : Joan Carter

原本協力者 : Robin J. Fisher, Frank Rovitto, and Philipp Weckerle, Shaun Lin, Vinay Pamadi, Rajesh Ramachandran, Danny Richardson, Jim Safcik, J. Toby Shimizu, Jeff Tang, and Vanessa Wang

Copyright © 1996, 2001, 2002, Oracle Corporation. All rights reserved.

Printed in Japan.

制限付権利の説明

プログラム（ソフトウェアおよびドキュメントを含む）の使用、複製または開示は、オラクル社との契約に記された制約条件に従うものとします。著作権、特許権およびその他の知的財産権に関する法律により保護されています。

当プログラムのリバース・エンジニアリング等は禁止されています。

このドキュメントの情報は、予告なしに変更されることがあります。オラクル社は本ドキュメントの無謬性を保証しません。

* オラクル社とは、**Oracle Corporation**（米国オラクル）または日本オラクル株式会社（日本オラクル）を指します。

危険な用途への使用について

オラクル社製品は、原子力、航空産業、大量輸送、医療あるいはその他の危険が伴うアプリケーションを用途として開発されておりません。オラクル社製品を上述のようなアプリケーションに使用することについての安全確保は、顧客各位の責任と費用により行ってください。万一かかる用途での使用によりクレームや損害が発生いたしましても、日本オラクル株式会社と開発元である **Oracle Corporation**（米国オラクル）およびその関連会社は一切責任を負いかねます。当プログラムを米国国防総省の米国政府機関に提供する際には、『**Restricted Rights**』と共に提供してください。この場合次の **Notice** が適用されます。

Restricted Rights Notice

Programs delivered subject to the DOD FAR Supplement are "commercial computer software" and use, duplication, and disclosure of the Programs, including documentation, shall be subject to the licensing restrictions set forth in the applicable Oracle license agreement. Otherwise, Programs delivered subject to the Federal Acquisition Regulations are "restricted computer software" and use, duplication, and disclosure of the Programs shall be subject to the restrictions in FAR 52.227-19, Commercial Computer Software - Restricted Rights (June, 1987). Oracle Corporation, 500 Oracle Parkway, Redwood City, CA 94065.

このドキュメントに記載されているその他の会社名および製品名は、あくまでその製品および会社を識別する目的にのみ使用されており、それぞれの所有者の商標または登録商標です。

目次

はじめに	xv
対象読者	xvi
このマニュアルの構成	xvi
関連文書	xvii
表記規則	xvii

第 I 部 環境の準備

1 Oracle9iAS Reports Services のアーキテクチャ

1.1	Oracle9iAS Reports Services の概要	1-2
1.2	Oracle9iAS Reports Services のコンポーネント	1-5
1.3	Oracle9iAS Reports Services のランタイム・プロセス	1-7
1.4	システム・セットアップ時の考慮事項	1-9
1.4.1	使用される要求タイプの選択	1-9
1.4.2	サーブレット、JSP、または CGI の選択	1-10
1.4.3	シングルマシン構成とマルチマシン構成の選択	1-10
1.4.4	複数サーバーのクラスタ化の選択	1-11

2 Oracle9iAS Reports Services の起動および停止

2.1	Reports Server の起動	2-2
2.1.1	サービスとしての Reports Server のインストールおよび起動 (Windows NT/2000)	2-2
2.1.2	サーブレットとしての Reports Server の起動 (Windows および UNIX)	2-3
2.1.3	コマンドラインからの Reports Server の起動 (Windows および UNIX)	2-3
2.2	Reports Servlet および Reports Server が実行されていることの確認	2-4

2.3	Oracle HTTP Server が実行されていることの確認	2-4
2.4	Reports Server の停止	2-5

3 Oracle9iAS Reports Services の構成

3.1	Oracle9iAS Reports Services 構成ファイル	3-2
3.2	Oracle9iAS Reports Server の構成	3-3
3.2.1	Reports Server の構成要素 (rwserverconf.dtd)	3-4
3.2.1.1	server	3-5
3.2.1.2	compatible	3-5
3.2.1.3	cache	3-7
3.2.1.4	engine	3-8
3.2.1.5	security	3-11
3.2.1.6	destination	3-12
3.2.1.7	job	3-14
3.2.1.8	notification	3-15
3.2.1.9	log	3-16
3.2.1.10	jobStatusRepository	3-17
3.2.1.11	trace	3-19
3.2.1.12	connection	3-21
3.2.1.13	queue	3-23
3.2.1.14	persistFile	3-24
3.2.1.15	identifier	3-24
3.2.1.16	pluginParam	3-25
3.3	Reports Servlet の構成	3-27
3.3.1	キー・マップ・ファイルの位置の指定	3-28
3.3.2	キー・マップ・ファイルの再ロード	3-28
3.3.3	冗長エラー・メッセージの非表示	3-29
3.3.4	ログイン・ダイアログ・ボックスの選択	3-29
3.3.5	Reports Servlet と JSP のトレース・オプションの設定	3-30
3.3.6	サーバー・エラー・メッセージの外観のカスタマイズ	3-31
3.3.7	インプロセス・サーバーの指定	3-31
3.3.8	デフォルトの Reports Server の識別	3-32
3.3.9	動的に生成されたイメージの指定	3-32
3.3.10	DB 認証と SYSAUTH Cookie の期限切れの設定	3-32
3.3.11	DB 認証 Cookie 用の暗号化キーの設定	3-33
3.3.12	診断 / デバッグ出力のフォーマットの追加	3-33

3.3.13	SSL ポート番号の指定	3-33
3.3.14	rwsservlet ヘルプ・ファイルの定義	3-33
3.3.15	シングル・サインオン使用の指定	3-34
3.4	URL エンジンの構成	3-34
3.5	プロキシ情報の入力	3-36
3.6	Oracle Enterprise Manager 用の Reports Server の構成	3-37

4 Oracle9iAS Reports Services の宛先の構成

4.1	出力処理の概要	4-2
4.2	サーバーへの宛先タイプの登録	4-4
4.2.1	サーバー構成ファイルの destination セクションの設定	4-4
4.2.2	宛先として有効な値の入力	4-5
4.2.2.1	宛先タイプとクラス	4-5
4.2.2.2	宛先プロパティの名前 / 値のペア	4-6

5 ユーザー・アクセスの制御

5.1	Oracle9iAS Portal の概要	5-2
5.2	サーバー構成ファイルにおけるポータルベースのセキュリティの定義	5-2
5.3	Oracle9iAS Portal での Reports ユーザーおよび名前付きグループの作成	5-4
5.3.1	デフォルトの Reports 関連グループ	5-5
5.3.1.1	RW_BASIC_USER	5-5
5.3.1.2	RW_POWER_USER	5-5
5.3.1.3	RW_DEVELOPER	5-5
5.3.1.4	RW_ADMINISTRATOR	5-6
5.3.2	ユーザーとグループの作成	5-6
5.4	アクセス制御の設定	5-6
5.4.1	使用可能日カレンダーの作成	5-7
5.4.1.1	シンプルな使用可能日カレンダーの作成	5-7
5.4.1.2	結合された使用可能日カレンダーの作成	5-9
5.4.2	プリンタの登録	5-11
5.4.3	Reports Server の登録	5-13
5.4.4	レポートの登録	5-15

6 Reports Server のクラスタ

6.1	クラスタの概要	6-2
-----	---------------	-----

6.2	クラスタのセットアップ	6-3
6.2.1	Reports Server 名の変更	6-3
6.2.2	公開鍵および秘密鍵の新規生成	6-5
6.2.3	公開鍵および秘密鍵のサーバー構成ファイルへの入力	6-5
6.2.4	Reports Server の再起動	6-6
6.2.5	クラスタへの要求送信	6-6

7 データ・ソース・シングル・サインオン

7.1	SSO のアーキテクチャ	7-3
7.1.1	SSO のコンポーネント	7-3
7.1.2	SSO のトランザクション	7-4
7.2	ユーザーの接続文字列を設定する方法	7-5
7.2.1	初期要件	7-5
7.2.2	方法 1: ユーザーに OID へのアクセスを許可する	7-6
7.2.3	方法 2: 接続文字列を割り当ててログイン時にユーザーに入力させる	7-6

第 II 部 サーバーへの要求送信

8 レポート要求の実行

8.1	Reports の URL 構文	8-2
8.1.1	サブレット	8-2
8.1.2	JSP	8-3
8.1.3	CGI	8-4
8.2	レポート要求の方法	8-5
8.3	Oracle9iAS Portal でのレポート・ポートレットの公開	8-7
8.3.1	レポートに対するプロバイダの作成	8-7
8.3.2	レポート定義ファイル・アクセスの作成	8-7
8.3.3	ページへのレポート・ポートレットの追加	8-8
8.4	Web ブラウザからのレポート要求の指定	8-9
8.5	URL エンジンへの要求送信	8-10
8.6	自動実行のためのレポートのスケジューリング	8-10
8.7	追加のパラメータ	8-11
8.8	キャッシュからのレポート出力の再使用	8-11
8.8.1	使用上の注意	8-12
8.9	キー・マップ・ファイルの使用方法	8-12

8.9.1	キー・マッピングを使用可能にする方法	8-13
8.9.2	キー・マップ・ファイルへのキー・マッピング・エントリの追加	8-14
8.9.3	JSP 以外のすべてでキーを使用する場合	8-14
8.9.4	JSP としてのレポート実行でのキーの使用方法	8-15

9 拡張配布の作成

9.1	配布の概要	9-2
9.2	配布 XML ファイルの概要	9-2
9.2.1	distribution.dtd ファイル	9-2
9.2.2	属性内での変数の使用に関する概略	9-3
9.3	配布 XML ファイルの要素	9-4
9.3.1	destinations	9-4
9.3.2	foreach	9-5
9.3.3	mail	9-7
9.3.4	body	9-10
9.3.5	attach	9-12
9.3.6	include	9-14
9.3.7	file	9-16
9.3.8	printer	9-18
9.3.9	destype	9-20
9.3.10	property	9-22
9.4	配布 XML ファイルの例	9-23
9.4.1	foreach の例	9-23
9.4.1.1	レポート・グループを個別の添付ファイルとする単一の電子メール	9-23
9.4.1.2	グループ・インスタンスごとに個別の電子メール	9-24
9.4.1.3	単独のセクションを添付ファイルとする個別の電子メール	9-24
9.4.1.4	セクションごとに異なるファイル	9-25
9.4.1.5	レポートごとに実行される印刷	9-25
9.4.1.5.1	Windows	9-25
9.4.1.5.2	UNIX	9-26
9.4.2	mail の例	9-26
9.4.2.1	レポート全体を本文とする電子メール	9-26
9.4.2.2	レポートの 1 つのセクションを本文とする電子メール	9-26
9.4.2.3	レポートの 2 つのセクションを本文とする電子メール	9-27
9.4.2.4	外部ファイルを本文とし、レポートを添付ファイルとする電子メール	9-27
9.4.2.4.1	Windows	9-27
9.4.2.4.2	UNIX	9-27

9.4.2.5	レポート全体およびグループ化されたセクションが添付された電子メール	9-28
9.4.2.6	関連する管理者と部門に送信される電子メール	9-28
9.4.3	file の例	9-29
9.4.3.1	レポート全体のファイル	9-29
9.4.3.1.1	Windows	9-29
9.4.3.1.2	UNIX	9-29
9.4.3.2	結合されたレポート・セクションのファイル	9-29
9.4.3.3	結合されたセクションから成る各グループのファイル	9-30
9.4.3.4	レポート・グループ・インスタンスごとのファイル	9-30
9.4.4	printer の例	9-30
9.4.4.1	レポート全体の印刷	9-31
9.4.4.1.1	Windows	9-31
9.4.4.1.2	UNIX	9-31
9.4.4.2	レポートの 2 つのセクションの印刷	9-31
9.4.4.2.1	Windows	9-31
9.4.4.2.2	UNIX	9-31
9.4.4.3	グループ化されたレポートの印刷	9-31
9.4.4.3.1	Windows	9-31
9.4.4.3.2	UNIX	9-32
9.4.4.4	結合されたセクションをグループ・インスタンスごとに印刷	9-32
9.4.4.4.1	Windows	9-32
9.4.4.4.2	UNIX	9-32
9.4.4.5	レポートの適切なインスタンスをその関連プリンタに印刷	9-33
9.5	実行時の配布 XML ファイルの使用方法	9-33
9.6	カスタム / Pluggable Destinations に対する XSL 変換	9-34

10 XML によるレポートのカスタマイズ

10.1	カスタマイズの概要	10-3
10.2	XML カスタマイズの作成	10-3
10.2.1	必須 XML タグ	10-4
10.2.2	スタイルの変更	10-4
10.2.3	書式マスクの変更	10-5
10.2.4	書式例外の追加	10-5
10.2.5	プログラム・ユニットとハイパーリンクの追加	10-6
10.2.6	新規問合せの追加と新規ヘッダー・セクションでの結果の使用	10-7
10.3	XML データ・モデルの作成	10-9
10.3.1	複数のデータ・ソースの作成	10-9

10.3.2	データ・ソース間のリンク	10-10
10.3.3	各データ・ソース内でのグループ階層の作成	10-11
10.3.4	クロス積（マトリックス）グループの作成	10-12
10.3.5	あらゆるレベルでの式、サマリーおよびプレースホルダの作成	10-13
10.3.6	パラメータの作成	10-14
10.4	実行時の XML ファイルの使用方法	10-15
10.4.1	実行時の XML レポート定義の適用	10-15
10.4.1.1	1 つの XML レポート定義の適用	10-16
10.4.1.2	複数の XML レポート定義の適用	10-16
10.4.1.3	PL/SQL での XML レポート定義の適用	10-17
10.4.1.3.1	ファイルに保存された XML 定義の適用	10-17
10.4.1.3.2	メモリーに保存された XML 定義の適用	10-17
10.4.2	XML レポート定義の単独での実行	10-20
10.4.3	バッチ変更の実行	10-21
10.5	XML レポート定義のデバッグ	10-21
10.5.1	XML パーサーのエラー・メッセージ	10-21
10.5.2	トレース・オプション	10-22
10.5.3	RWBUILDER	10-22
10.5.4	デバッグを目的とした、ファイルへの XML の書込み	10-23

11 イベント・ドリブンによる公開

11.1	Event-Driven Publishing API	11-2
11.1.1	API の要素	11-2
11.1.2	パラメータ・リストの作成と操作	11-3
11.1.2.1	Add_Parameter	11-3
11.1.2.2	Remove_Parameter	11-4
11.1.2.3	Clear_Parameter_List	11-4
11.1.3	ジョブの送信方法	11-4
11.1.4	ステータスの確認方法	11-6
11.1.5	サーバーのステータス・レコードの使用方法	11-6
11.2	Event-Driven Publishing API を使用するアプリケーションのデバッグ	11-7
11.3	データベース・イベントからのレポートの起動	11-8
11.4	Oracle9i Advanced Queuing との統合	11-10
11.4.1	SRW_PARAMLIST タイプのメッセージを保持するキューの作成	11-10
11.4.2	エンキュー・プロシージャの作成	11-11
11.4.3	デキュー・プロシージャの作成	11-12

第 III 部 各国語サポートおよび双方向サポート

12 NLS と双方向サポート

12.1	NLS アーキテクチャ	12-2
12.1.1	言語に依存しない機能	12-2
12.1.2	言語依存データ	12-2
12.2	NLS の環境変数	12-2
12.2.1	NLS_LANG 環境変数	12-3
12.2.1.1	NLS_LANG 環境変数の定義	12-6
12.2.1.1.1	Windows	12-6
12.2.1.1.2	UNIX	12-6
12.2.1.2	キャラクタ・セット	12-6
12.2.1.2.1	キャラクタ・セットの設計上の考慮事項	12-6
12.2.1.2.2	Windows プラットフォームでのフォントのエイリアシング	12-7
12.2.1.3	言語と地域	12-8
12.2.2	DEVELOPER_NLS_LANG 環境変数と USER_NLS_LANG 環境変数	12-9
12.3	JSP または XML ファイルでのキャラクタ・セットの指定	12-9
12.4	双方向サポート	12-12
12.5	Unicode	12-13
12.5.1	Unicode のサポート	12-13
12.5.2	Unicode フォントのサポート	12-14
12.5.3	Unicode サポートの有効化	12-14
12.5.4	ALTER SESSION の使用方法	12-15
12.6	アプリケーションの翻訳	12-15

第 IV 部 パフォーマンス

13 Oracle9/AS Reports Services の管理と監視

13.1	OEM での Reports Services 情報へのナビゲート	13-3
13.2	Reports Server の開始、停止および再起動	13-3
13.3	Reports ジョブ・キューの表示と管理	13-4
13.3.1	現行ジョブ・キューの表示と管理	13-4
13.3.1.1	Reports Server の現行ジョブ・キューの表示	13-4
13.3.1.2	現行ジョブの取消し	13-5
13.3.2	スケジュールされたジョブ・キューの表示と管理	13-5

13.3.2.1	Reports Server のスケジュールされたジョブ・キューの表示	13-6
13.3.2.2	スケジュールされたジョブの取消し	13-6
13.3.3	終了したジョブ・キューの表示と管理	13-7
13.3.3.1	Reports Server の終了したジョブ・キューの表示	13-7
13.3.3.2	ジョブのトレース・ファイルの表示	13-9
13.3.3.3	キャッシュからの結果の表示	13-9
13.3.3.4	終了したジョブの再実行	13-9
13.3.4	失敗したジョブ・キューの表示と管理	13-9
13.3.4.1	Reports Server の失敗したジョブ・キューの表示	13-9
13.3.4.2	失敗したジョブのトレース・ファイルの表示	13-11
13.3.4.3	失敗したジョブの再実行	13-11
13.4	サーバー・パフォーマンスの監視	13-11
13.5	Reports Server 構成ファイルの表示と変更	13-12
13.6	サーバー・クラスタ・メンバーの表示とメンバーへのリンク	13-13
13.7	OEM への Reports Server の追加	13-14

14 Oracle9iAS Reports Services のチューニング

14.1	インプロセス・サーバーの使用方法	14-2
14.2	Reports Engine のチューニング	14-2
14.2.1	initEngine	14-3
14.2.2	maxEngine	14-3
14.2.3	minEngine	14-4
14.2.4	engLife	14-4
14.2.5	maxIdle	14-4
14.2.6	callBackTimeOut	14-5
14.3	複数サーバーのクラスタ化	14-5
14.4	キャッシュ方法の最適化	14-6
14.4.1	Reports Server 構成ファイルでのキャッシュの設定	14-7
14.4.2	コマンドラインでのキャッシュ関連オプションの指定	14-8
14.4.2.1	TOLERANCE	14-8
14.4.2.2	EXPIRATION	14-8
14.4.3	JSP でのキャッシュ・オプションの設定	14-9
14.5	パフォーマンスの監視	14-10
14.5.1	Oracle Trace によるパフォーマンスの監視	14-11
14.5.1.1	Trace の概要	14-11
14.5.1.2	トレース情報のその他のソース	14-12
14.5.2	SHOWJOBS コマンド・キーワード	14-13

14.5.3	RW_SERVER_QUEUE 表へのアクセス	14-14
14.5.4	キュー・アクティビティによるデータベースの更新	14-17

第 V 部 付録

A コマンドライン引数

A.1	コマンドの概要	A-2
A.1.1	rwclient	A-2
A.1.2	rwrun	A-3
A.1.3	rwbuilder	A-3
A.1.4	rwconverter	A-4
A.1.5	rwervlet	A-5
A.1.6	rwcgi	A-6
A.1.7	rwserver	A-6
A.2	コマンドライン構文	A-7
A.3	全般的な使用上の注意	A-7
A.4	コマンドライン引数	A-7
A.4.1	ACCESSIBLE	A-7
A.4.2	ARRAYSIZE	A-8
A.4.3	AUTHID	A-8
A.4.4	AUTOCOMMIT	A-9
A.4.5	AUTOSTART	A-9
A.4.6	BATCH	A-10
A.4.7	BCC	A-10
A.4.8	BLANKPAGES	A-11
A.4.9	BUFFERS	A-12
A.4.10	CACHELOB	A-12
A.4.11	CC	A-13
A.4.12	CELLWRAPPER	A-13
A.4.13	CMDFILE	A-14
A.4.14	CMDKEY	A-15
A.4.15	CONTENTAREA	A-16
A.4.16	COPIES	A-17
A.4.17	CUSTOMIZE	A-18
A.4.18	DATEFORMATMASK	A-18
A.4.19	DELAUTH	A-19
A.4.20	DELIMITED_HDR	A-20

A.4.21	DELIMITER	A-20
A.4.22	DESFORMAT	A-21
A.4.23	DESNAME	A-22
A.4.24	DEST	A-23
A.4.25	DESTINATION	A-24
A.4.26	DESTYPE	A-25
A.4.27	DISTRIBUTE	A-26
A.4.28	DTYPE	A-27
A.4.29	DUNIT	A-28
A.4.30	EXPIRATION	A-29
A.4.31	EXPIREDAYS	A-30
A.4.32	EXPRESS_SERVER	A-30
A.4.33	FORMSIZE	A-32
A.4.34	FROM	A-33
A.4.35	GETJOBID	A-34
A.4.36	GETSERVERINFO	A-34
A.4.37	HELP	A-35
A.4.38	IGNOREMARGIN	A-35
A.4.39	INSTALL	A-36
A.4.40	ITEMTITLE	A-36
A.4.41	JOBNAME	A-37
A.4.42	JOBTYPE	A-37
A.4.43	KILLJOBID	A-38
A.4.44	LONGCHUNK	A-38
A.4.45	MODE	A-39
A.4.46	MODULE REPORT	A-40
A.4.47	NONBLOCKSQL	A-40
A.4.48	NOTIFYFAILURE	A-41
A.4.49	NOTIFYSUCCESS	A-41
A.4.50	NUMBERFORMATMASK	A-42
A.4.51	ONFAILURE	A-42
A.4.52	ONSUCCESS	A-43
A.4.53	ORIENTATION	A-44
A.4.54	OUTPUTFOLDER	A-44
A.4.55	OUTPUTPAGE	A-45
A.4.56	OVERWRITE	A-46
A.4.57	P_AVAILABILITY	A-47
A.4.58	P_DESCRIPTION	A-47

A.4.59	P_FORMATS	A-48
A.4.60	P_NAME	A-48
A.4.61	P_OWNER	A-49
A.4.62	P_PFORMTEMPLATE	A-49
A.4.63	P_PRINTERS	A-50
A.4.64	P_PRIVILEGE	A-50
A.4.65	P_SERVERS	A-51
A.4.66	P_TRIGGER	A-51
A.4.67	P_TYPES	A-52
A.4.68	PAGEGROUP	A-52
A.4.69	PAGESIZE	A-53
A.4.70	PAGESTREAM	A-54
A.4.71	PARAMFORM	A-55
A.4.72	PARSEQUERY	A-55
A.4.73	PDFCOMP	A-56
A.4.74	PDFEMBED	A-56
A.4.75	PRINTJOB	A-57
A.4.76	READONLY	A-57
A.4.77	REPLACEITEM	A-58
A.4.78	REPLYTO	A-59
A.4.79	REPORT MODULE	A-59
A.4.80	ROLE	A-59
A.4.81	RUNDEBUG	A-60
A.4.82	SAVE_RDF	A-60
A.4.83	SCHEDULE	A-61
A.4.84	SERVER	A-62
A.4.85	SHOWENV	A-63
A.4.86	SHOWJOBS	A-63
A.4.87	SHOWMAP	A-64
A.4.88	SHOWMYJOBS	A-64
A.4.89	SHUTDOWN	A-65
A.4.90	SITENAME	A-65
A.4.91	SOURCE	A-66
A.4.92	SSOCONN	A-67
A.4.93	STATUSFORMAT	A-69
A.4.94	STATUSFOLDER	A-69
A.4.95	STATUSPAGE	A-70
A.4.96	STYPE	A-71

A.4.97	SUBJECT	A-71
A.4.98	TOLERANCE	A-72
A.4.99	TRACEFILE	A-73
A.4.100	TRACEMODE	A-74
A.4.101	TRACEOPTS	A-74
A.4.102	UNINSTALL	A-76
A.4.103	URLPARAMETER	A-76
A.4.104	USERID	A-77
A.4.105	WEBSERVER_DEBUG	A-77
A.4.106	WEBSERVER_DOCROOT	A-78
A.4.107	WEBSERVER_PORT	A-79

B Reports 関連の環境変数

C Oracle9iAS Portal へのレポートのバッチ登録

C.1	レポート定義ファイルのバッチ登録	C-2
C.1.1	RWCONVERTER の実行による SQL スクリプトの生成	C-2
C.1.2	SQL*Plus でのスクリプトの実行	C-4
C.2	レポート・パッケージのバッチ削除	C-5
C.3	PL/SQL バッチ登録ファンクション	C-5

索引

はじめに

このマニュアルでは、レポートを公開するための Oracle9iAS Reports Services ソフトウェアの構成方法について説明します。また、Oracle9iAS Reports Services を使用してレポートを公開するために用意されている各種オプションについても説明します。

対象読者

このマニュアルは、Oracle9iAS Reports Services を使用して、レポートを公開する方を対象に記述されています。Oracle9iAS Reports Services を構成する場合は、次のテクノロジーについて確実に理解しておくに役立ちます。

- 使用しているオペレーティング・システム
- Java
- データベース
- CORBA
- JSP ファイル
- XML ファイルおよび DTD ファイル
- Web サーバーの構成
- HTTP

このマニュアルでは、これらのテクノロジーに関するコンポーネントを構成する手順を説明します。

このマニュアルの構成

このマニュアルには、次の章が含まれています。

- 第 1 章 Oracle9iAS Reports Services アーキテクチャの概要について説明します。
- 第 2 章 Oracle9iAS Reports Services の開始および停止方法について説明します。
- 第 3 章 Oracle9iAS Reports Services の構成方法について説明します。
- 第 4 章 Oracle9iAS Reports Services でデフォルトの宛先やカスタムの宛先に対し出力が処理される方法について説明します。
- 第 5 章 Oracle9iAS Portal を使用して、レポートに対するユーザー・アクセスを制御する方法について説明します。
- 第 6 章 Reports Server をクラスタ化して、パフォーマンスおよび信頼性を強化する方法について説明します。
- 第 7 章 シングル・サインオン・アーキテクチャの概要と、処理フローについて説明し、ユーザーのリソース情報を Oracle Internet Directory で取得する際のヒントも提供します。
- 第 8 章 レポートの様々な実行方法、特にランタイム URL の構成方法について説明します。
- 第 9 章 配布 XML ファイルを使用して拡張配布を設定する方法について説明します。

第 10 章	XML を使用して、実行時のレポートのカスタマイズを行う方法について説明します。
第 11 章	イベント・ドリブンの公開を使用して、データベース・トリガーに応じてレポートを自動的に起動する方法について説明します。
第 12 章	Oracle9iAS Reports Services に関する NLS の情報を提供します。
第 13 章	Oracle9iAS Reports Services と Oracle Enterprise Manager (OEM) との統合、および Reports Server と OEM を監視する方法について説明します。
第 14 章	チューニングおよびパフォーマンス強化のヒントを提供します。
付録 A	Reports に関連するコマンドライン引数の情報を提供します。
付録 B	Oracle9iAS Reports Services の環境変数に関する情報を提供します。
付録 C	バッチ処理でレポートを Oracle9iAS Portal に登録する方法に関する情報を提供します。

関連文書

レポートの作成、Oracle9iAS Portal、または Oracle9iAS Reports Services の詳細は、次のマニュアルを参照してください。

- 『Oracle9i Reports チュートリアル』
- 『Oracle9i Reports レポート作成ガイド』
- 『Oracle9i Reports Developer スタート・ガイド』

表記規則

このマニュアルでは、次のような表記規則を使用しています。

表記規則	意味
太字	強調を目的として使用。メニュー項目、ボタン名、ラベルなどユーザー・インタフェース要素にも使用。
固定幅フォント	パスおよびファイル名、入力するコードおよびテキストに使用。
大文字	環境変数、コマンドライン・キーワード、Windows NT プラットフォームでのビルトイン名およびパッケージ名に使用。UNIX プラットフォームでは小文字を使用。

第I部

環境の準備

第I部では、Oracle9iAS Reports Services 環境の概要を説明し、レポート実行に向けその環境を準備する際に役立つ情報を提供します。これには、Oracle9iAS Reports Services の起動と停止、Reports 関連の Oracle9iAS コンポーネントの構成、Oracle9iAS Portal を使用した実行環境の保護、複数の Reports Server のクラスタ化およびデータ・ソース・シングル・サインオンの設定などの情報が含まれます。

第I部には、次の章があります。

- 第1章「Oracle9iAS Reports Services のアーキテクチャ」
- 第2章「Oracle9iAS Reports Services の起動および停止」
- 第3章「Oracle9iAS Reports Services の構成」
- 第4章「Oracle9iAS Reports Services の宛先の構成」
- 第5章「ユーザー・アクセスの制御」
- 第6章「Reports Server のクラスタ」
- 第7章「データ・ソース・シングル・サインオン」

Oracle9iAS Reports Services の アーキテクチャ

レポートを公開する準備が整ったら、必要な Web サーバーおよびアプリケーション・サーバー・ツールはすべて Oracle9i Application Server (Oracle9iAS) で利用できます。この章では、関連する Oracle9iAS コンポーネントのアーキテクチャをレポート公開コンポーネント (Oracle9iAS Reports Services) とともに説明します。また、レポート実行時処理の概要についても説明し、サーバー環境セットアップ時の考慮事項も示します。

この章には、次の項があります。

- [Oracle9iAS Reports Services の概要](#)
- [Oracle9iAS Reports Services のコンポーネント](#)
- [Oracle9iAS Reports Services のランタイム・プロセス](#)
- [システム・セットアップ時の考慮事項](#)

1.1 Oracle9iAS Reports Services の概要

Oracle9iAS は、各種 Web サイト、ポータルまたはインターネット・アプリケーションを実行する包括的な統合アプリケーション・サーバーです。このサーバーにより、アプリケーションを Web ブラウザ、携帯情報端末およびコマンドラインから利用できます。

Oracle9iAS Reports Services は、Oracle9iAS のレポート公開コンポーネントです。これは、質の高い運用レポートを作成する企業レポート・サービスで、動的にあらゆるデータを取得し、あらゆるフォーマットに成形し、そしてあらゆる宛先へ配布します。Oracle9iAS Reports Services を使用すると、Web ベースの環境と非 Web ベースの環境の両方での公開が可能です。

拡張性に優れ柔軟性の高いアーキテクチャを備える Oracle9iAS Reports Services は、レポート生成エンジンを同一のサーバーおよび複数のサーバーに分散し、自動的に管理します。また、同様の要求が発行された際に再利用できるよう、レポート出力をキャッシュします。このサービスは、JSP、Java Servlet および CGI を使用した標準の Web 環境と統合できます。また、ローカルとリモートの両アプリケーション・サーバーでのレポート実行、およびレポート実行用に複数層アーキテクチャの実装が可能です。

Oracle9iAS Reports Services を、サーブレット、JSP または CGI（下位互換性を確保するためにのみ維持）と組み合わせて使用する場合、標準的な URL 構文を使用すると任意のプラットフォームの Web ブラウザからレポートを実行できます。サーブレットを実装する場合、応答が速く、管理が簡単なインプロセス・サーバーを使用できます。インプロセス・サーバーを使用すると、プロセス間の通信コストが削減され、結果的に応答時間が短縮されます。

Oracle9iAS Reports Services は、すべてのジョブ要求を 1 つのジョブ・キューに入れてレポートを実行するようクライアント要求を操作します。いずれかのサーバーのエンジンが使用可能になると、キューで待機している次のジョブが実行されます。キューのジョブ数が増えると、サーバーはエンジンをサーバー構成で指定した最大数まで増やすことができます。同様に、エンジンは、指定したアイドル時間（第 3 章「[Oracle9iAS Reports Services の構成](#)」を参照）を超えた場合、自動的に停止します。

Oracle9iAS Reports Services は、実行中のジョブ、実行予定のジョブ、完了済のジョブまたは失敗したジョブなど、サーバーに送信されたすべてのジョブを追跡します。Reports Queue Manager (Windows)、Reports Queue Viewer (UNIX)、showjobs コマンド (Web)、および Oracle Enterprise Manager (OEM) の Reports Services ページを使用して、ジョブのスケジューリング、キューイング、開始、終了およびエラーの時刻だけでなく、ジョブ出力とレポートの最終ステータスに関する情報を表示できます。

Oracle9iAS Reports Services のジョブ・オブジェクトは永続的です。したがって、サーバーを停止した後には再起動すると、¹ スケジュールされたジョブだけでなく、すべてのジョブが回復します。

¹ この場合、同期ジョブおよび現在実行中のジョブのみが消失します。

Web 環境で使用する場合、Oracle9iAS Reports Services のアーキテクチャは次の 4 つの階層で構成されます。

注意： 階層という用語は、Oracle9iAS Reports Services のアーキテクチャに準ずる構成要素の論理的位置を意味します。しかしながら、それぞれの階層は、同一マシン上にも、異なるマシン上にも存在することがあります。

- クライアント層 (Web ブラウザ)
- Web サーバー層
- Oracle9iAS Reports Services 層
- データ層 (データベースおよびその他すべてのデータ・ソースを含む)

非 Web 環境で使用する場合は、次の 3 つの階層で構成されます (Web サーバーが不要になります)。

- クライアント層 (クライアント・マシンごとに、小さな独自アプリケーション)
- Oracle9iAS Reports Services 層
- データ層 (データベースおよびトランспортаブル・データ・ソースを含む)

これらの層の構成方法は、すべての階層を同一マシン上に設定する方法や各階層をすべて別のマシン上に設定する方法などがあります。また、複数のマシンに複数の Web サーバーをインストールしたり、複数のマシンに複数のアプリケーション・サーバーをインストールすることもできます。

Web サーバーを複数のマシンにインストールすると、Oracle9iAS HTTP Server ではロード・バランシング機能が提供され、Web サーバーの負荷を複数のマシンに分散できます。アプリケーション・サーバーを複数のマシンにインストールすると、Oracle9iAS Reports Services でピア・レベルのクラスタ化が可能になり、Reports Server の負荷を複数のマシンに分散できます。

ロード・バランシングとピアレベルのクラスタ化の違いは、ロード・バランシングでは 1 台のマシンですべてのマシンの通信量を管理するのに対し、ピアレベルのクラスタ化ではすべてのマシンが各マシン上の通信量を監視し、各マシンで監視タスクおよび要求への応答タスクを共有することです。ピアレベルのクラスタ化の利点は、単一の障害箇所を排除し、フェイルセーフ環境をさらに強化することです。

注意： Reports Server のクラスタ化の詳細は、[第 6 章「Reports Server のクラスタ」](#)を参照してください。

Oracle9iAS Reports Services では、イベント・ベースのレポートを提供します。この方法では、データベース・イベントを使用して、レポート生成を実行します。たとえば、収益レベルが特定の水位標を上回った（下回った）という変更を通知するイベントを定義できます。データベースでこの変更（イベント）が発生すると、レポートが自動的に生成されます。詳細は、[第 11 章「イベント・ドリブンによる公開」](#)を参照してください。

Oracle9iAS Reports Services には、XML を使用して、レポート配布用の独自の構成を定義する配布モジュールが付属しています。必要な XML ファイルをランタイム・コマンドラインまたは URL から呼び出してレポートを 1 つ生成すれば、多様な出力と宛先の処理がサーバーにより実行されます。処理時間が大幅に短縮され、構成変更はすべて XML ファイル内で処理できます。詳細は、[第 9 章「拡張配布の作成」](#)を参照してください。

1.2 Oracle9iAS Reports Services のコンポーネント

図 1-1 Oracle9iAS Reports Services のコンポーネント

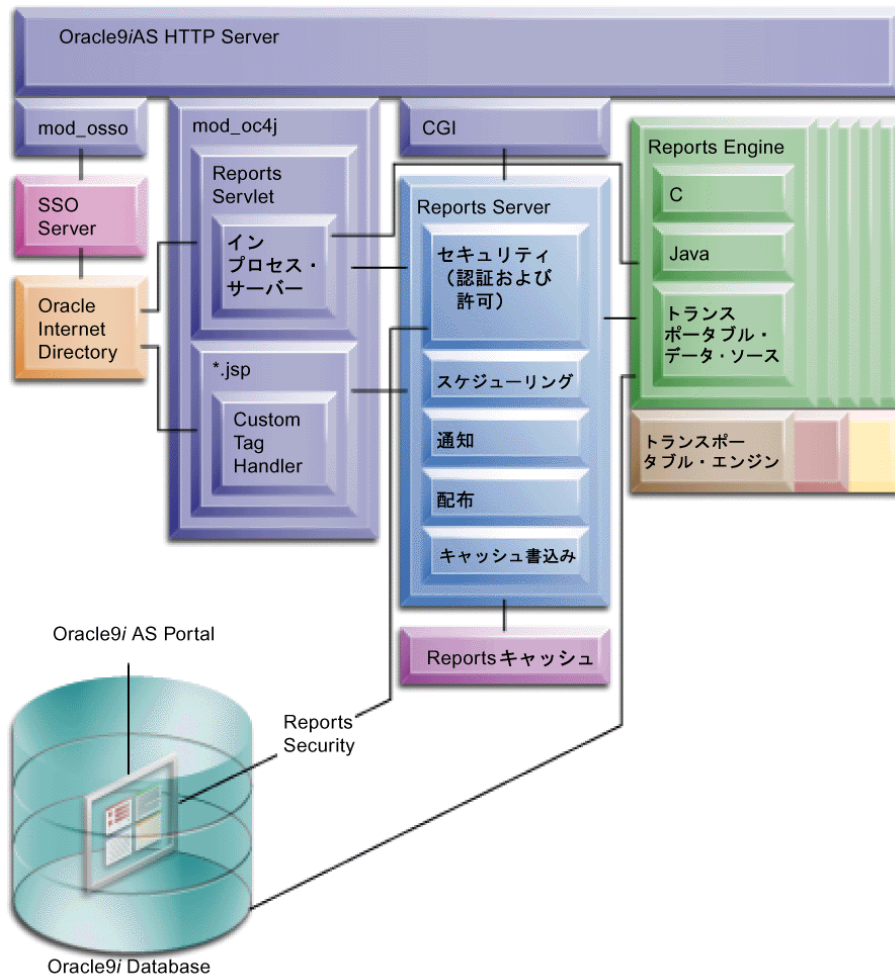


図 1-1 は、稼働中の Oracle9iAS Reports Services 環境における次のコンポーネントを示しています。

1. Oracle9iAS HTTP Server:

Oracle9iAS は、Oracle9iAS HTTP Server という Web サーバーを提供します。組み込みの OpenSSL モジュールを持つこのサーバーは、Secure Sockets Layer (SSL) および HTTP Secure Sockets Layer (HTTPS) をサポートします。また、Java Servlet アプリケーションをサポートするサーブレット・エンジンも備えています。

2. モジュール **mod_oc4j**。Oracle9iAS HTTP Listener で使用するこのモジュールは、要求をサーブレットおよび JSP から Oracle9iAS Containers for Java 2 Enterprise Edition (OC4J) にリダイレクトします。OC4J は、JSP トランスレータ、JSP サーブレット・エンジン (OJSP) および Enterprise JavaBeans (EJB) コンテナを備える完全な J2EE 環境を提供します。OC4J は、高速、軽量、高拡張性、簡易性を兼ね備えた完全な J2EE 環境を提供します。すべて Java で記述されている OC4J は、標準の Java Development Kit (JDK) 仮想マシン (JVM) 上で動作します。
3. Oracle9iAS Reports Services のコンポーネントである **Reports Servlet** は、Web サーバーのサーブレット・エンジン内で動作し、HTTP Server と Reports Server との間で情報を変換および配布します。Reports Servlet は、次のコンポーネントで構成されています。
 - **インプロセス・サーバー**: クライアントからの最初の要求を Reports Servlet (rwservlet) または Reports JSP 経由で受信するたびに、サーバーを自動的に起動する手段を提供するので、Reports Server のメンテナンスと管理を軽減します。
 - **Custom Tag Handler**: JSP ファイルに含まれるカスタム Reports タグを処理します (JSP ファイルでは、Reports 関連のカスタム・タグは接頭辞が **rw:** です。その他の接頭辞を持つその他のカスタム・タグが存在することもあります)。
4. Web サーバーの **Reports CGI** コンポーネントは、HTTP Server と Reports Server の間で情報を変換および配布します。

注意: Reports CGI は、下位互換性を確保するためにのみ維持されています。

5. **Reports Server** は、クライアント要求を処理します。この処理では、クライアント要求を、認証と許可の確認、スケジューリング、キャッシュおよび配布 (カスタムまたは Pluggable Destination への配布など) など、各種サービスに割り当てます。また、要求されたレポートを生成するランタイム・エンジンを起動したり、生成されたレポートを Reports キャッシュから取り出したり、ジョブの準備が完了したことをクライアントに通知したりします。
6. **Reports キャッシュ**では、完了済のジョブ出力が安全に格納されます。
7. **Reports Engine** は、SQL ベースおよびトランспортаブル・データ・ソースベースのレポートを実行するコンポーネントで構成されています。Reports Engine は、要求されたデータをデータ・ソースから取り出したり、レポートをフォーマットしたり、出力をキャッシュに送信したり、Reports Server にジョブ完了通知を行ったりします。

8. トランスポータブル・エンジンは、Java API を使用してジョブを Reports Server に渡すほか、スケジューリング、配布、通知およびキャッシュなどサーバーのその他の機能を利用するカスタム・エンジンです。Oracle9iAS Reports Services は、URL エンジンと呼ばれる、特別な設定を必要としないトランスポータブル・エンジンを提供します。URL エンジンを使用すると、任意の公開 URL のコンテンツを電子メール、Oracle9iAS Portal および WebDav などの宛先に配布できます。

1.3 Oracle9iAS Reports Services のランタイム・プロセス

ここでは、Oracle9iAS Reports Services の各種コンポーネントがレポートの実行プロセスで果たす役割について説明します。

1. クライアントは、URL (Web) を使用するか、Web を使用せずに Reports 関連のコマンド (rwclient など) を使用してサーバーに接続し、レポートを要求します。
 - URL を使用した要求は、JSP、rwservlet または CGI にリダイレクトされます。これらはすべて Oracle HTTP Listener に接続されています。JSP と rwservlet の要求は、mod_oc4j にリダイレクトされます (ジョブは JSP として実行されるので、mod_oc4j は OJSP を使用して JSP をサーブレットに変換します)。CGI 要求は、CGI コンポーネントにリダイレクトされます。

URL には、ランタイム・パラメータまたは cgicmd.dat 内のランタイム・パラメータ構成セクションを参照するキーワードを指定できます。両方指定することもできますが、URL で明示的に指定されているパラメータが cgicmd.dat の関連キーワード・セクションに存在してはなりません。
 - rwclient を使用した要求は、直接 Reports Server にリダイレクトされます。

コマンドラインには、ランタイム・パラメータを指定できます。ランタイム・パラメータが多数ある場合には、rwclient コマンドとパラメータ文字列を含むバッチ・ファイルまたはシェル・スクリプトを作成できます。
2. rwservlet または Reports CGI (rwcgi は下位互換性を確保するためにのみ維持) コンポーネントは、HTTP Server と Reports Server 間で要求を変換し、要求を Reports Server に送信します。
 - JSP または rwservlet を使用したサーバー要求は、インプロセス Reports Server で実行されるか、スタンドアロンの Reports Server プロセスとして実行されます。どちらであるかは、サーブレットの構成ファイル rwservlet.properties (%ORACLE_HOME%\reports\conf¥) の指定により異なります。インプロセス・サーバーは、スタンドアロン・サーバーと異なり、クライアントからの要求に応答して自動的に起動するため、メンテナンスが軽減されます。また、インプロセス・サーバーを使用するとプロセス間の通信が削減されるため、パフォーマンスが向上します。
 - rwcgi を使用したサーバー要求は、スタンドアロン・サーバーにリダイレクトされます。
3. Reports Server が要求を処理します。

要求に TOLERANCE 引数が含まれる場合、Reports Server はキャッシュをチェックして、その要求を満たす既存の出力があるかどうかを判別します。キャッシュ内に要求を満たす出力があった場合、サーバーはレポートの再実行よりも、その出力を返すことを優先して行います。

注意： Reports Server に送るジョブ要求はすべて、許容値を示す引数 (tolerance) を指定できます。許容値は、要求者が許容できる最も古い出力を定義します。たとえば、要求者が許容値として 5 分を設定した場合、Reports Server はキャッシュをチェックして過去 5 分以内に作成された最新の複製レポート出力を探します。expiration 引数は、レポート出力をキャッシュから削除する日時を定義します (たとえば、出力の期限切れと同じ日時を指定できます)。詳細は、[付録 A「コマンドライン引数」](#)を参照してください。

要求が現在実行中のジョブと同じ場合、その要求は、レポートを再実行せずに現在のジョブの出力を再利用します。

これらの条件が満たされない場合、Reports Server は要求を処理します。

- a. Reports Server は、Oracle HTTP Server の一部である mod_osso を介して認証および許可の確認を開始します (構成されている場合)。
- b. レポートがスケジュールされていれば、Reports Server は要求をスケジュールされたジョブ・キューに挿入し、レポートはスケジュールに従って実行されます。レポートがスケジュールされていなければ、要求は現在のジョブ・キューに挿入され、Reports Engine が利用可能になった時点で実行されます。

注意： Reports Server を構成する際に (<server_name>.conf)、使用可能な Oracle9iAS Reports Engine の最大数を指定できます。Oracle9iAS Reports Server が指定した最大数を下回る場合は、ジョブをアイドル・エンジンに送信したり、新しいエンジンを開始して、要求を処理できます。そうでない場合は、現在の Oracle9iAS Reports Engine が現在のジョブを完了するまで、要求の実行を待つ必要があります。

- c. 実行時、Reports Server は Reports Engine を起動し、実行対象の要求をそのエンジンに送ります。
4. Reports Engine は、データを取り出し、フォーマットします。
5. Reports Engine は、Reports キャッシュに書き込みます。
6. Reports Engine は、レポートの準備が完了したことを Reports Server に通知します。
7. Reports Server は、URL、コマンドラインまたは cgicmd.dat ファイルのキーワード・セクション (URL 要求のみ) で指定されたランタイム・パラメータに従って、キャッシュにアクセスし、出力するレポートを送ります。

また、イベント・ドリブンの公開を使用してレポートを作成することもできます。イベント・ドリブンの公開では、クライアントはエンドユーザーではなくデータベースです。イベントは、Event-Driven Publishing API で定義されます。イベントは、Event-Driven Publishing API を呼び出すデータベース・トリガー、アドバンスド・キューイング・アプリケーションまたは PL/SQL パッケージを起動して、ジョブを Reports Server に送ります。イベント・ドリブンの公開の詳細は、第 11 章「イベント・ドリブンの公開」を参照してください。

1.4 システム・セットアップ時の考慮事項

Oracle9iAS Reports Services のセットアップ方法は、システムへの要求事項に応じて大きく異なります。Oracle9iAS Reports Services をセットアップする前に、要求事項に応じて決定する必要があります。事前にそのような決定を行うと、セットアップ・プロセスは非常に簡単になります。

次の各項では、これらの決定事項について説明します。

- 使用される要求タイプの選択
- サブレット、JSP、または CGI の選択
- シングルマシン構成とマルチマシン構成の選択
- 複数サーバーのクラスタ化の選択

1.4.1 使用される要求タイプの選択

Oracle9iAS Reports Services は、Web ベースの要求と非 Web ベースの要求の両方を受け入れるよう構成できます。

Web ベースの場合、Web ブラウザで URL をクリックするか入力することでレポートを実行できます。URL に従って、レポート出力はブラウザに返されるか、プリンタなど指定された宛先に送信されます。ユーザーがブラウザからレポートを起動できるようにするには、Oracle9iAS Reports Servlet、JSP または Oracle9iAS Reports CGI のいずれかのコンポーネントを Web サーバーで使用します。これらのコンポーネントのうち 1 つと Oracle9iAS Reports Server が通信し、Web クライアントからのレポート要求の処理を可能にするには、そのコンポーネントが Web サーバー上に置かれている必要があります。

注意： 詳細は、1.4.2 項「サブレット、JSP、または CGI の選択」を参照してください。

非 Web ベースの場合、各ユーザー・マシンにインストールされている `rwclient` 実行可能ファイルを使用して、ジョブ要求を送信できます。

構成の観点から見た、Web 要求と非 Web 要求の有効化の重要な相違点は、次のとおりです。

- Web 要求を有効にする場合、Web サーバー側で Reports Servlet、JSP または Reports CGI（下位互換性を確保するためにのみ維持）を選択する必要がありますが、標準 Web ブラウザ以外にクライアント・ソフトウェアをインストールする必要はありません。
- 非 Web 要求を有効にする場合、要求の実行に使用するクライアント・ソフトウェアを各マシンにインストールする必要があります。このため、各クライアント・マシンでクライアント・ソフトウェアを管理する必要があります。

Web ベースにすると、クライアントのメンテナンス・コストが削減されるため、最も費用効果が高くなります。ただし、非 Web 要求の実行が必要な場合もあります。Oracle9iAS Reports Services は、単一の実行環境で Web 要求と非 Web 要求の両方の実装をサポートします。

1.4.2 サブレット、JSP、または CGI の選択

Oracle9iAS Reports Services を Web 環境で使用するには、サブレット、JSP、または CGI を実装する必要があります。サブレットまたは JSP の選択をお勧めします。Oracle9iAS Reports Services での CGI の実装は、下位互換性を確保するためにのみ維持されています。

サブレットと JSP のどちらを選択するかについては、さらに考慮事項が必要です。JSP のみを実装すると、Web 配信に適したレイアウトを公開できます。サブレットを使用すると、ペーパー・レイアウトをレポート公開ソリューションに含めることができ、Oracle9iAS Reports Services の配布機能を最大限に活用できます。

JSP ファイルには Web レイアウトとペーパー・レイアウトの両方を含めることができるため、サブレットを使用しても JSP ファイルも使用できます。JSP に格納されているレポートを実行するときは、サブレットを URL で指定し、JSP をコマンドライン引数（report=<myreport>.jsp）で呼び出します。

レポート実行の詳細は、[第 8 章「レポート要求の実行」](#)を参照してください。

1.4.3 シングルマシン構成とマルチマシン構成の選択

Oracle9iAS Reports Services は、Web サーバーと同じマシン上に置くことも、異なるマシン上に置くこともできます。どちらも長所と短所があります。

たとえば、Oracle9iAS Reports Services と Web サーバーを同じマシンに置くと、マシンのメモリとディスク領域の消費量が増えますが、ネットワークの通信量は減少します。これは、Web サーバーとアプリケーション・サーバー間でやり取りする要求がネットワークを経由せず、受信要求のみがネットワークを経由するためです。

インプロセス・サーバーを使用している場合（サブレットを実装している場合のみ使用可能）、シングルマシンのパフォーマンス上の利点がさらに高まります。インプロセス・サーバーは、Oracle9iAS Reports Services コンポーネント間の通信をより高速化および効率化するので、処理時間が短縮されます。Reports Servlet を使用せずにレポートを配布する場合を除き、インプロセス・サーバーを使用することをお勧めします。

一方、シングルマシン構成を使用している場合に、そのマシンに障害が発生すると、すべてが失敗します。

Web サーバーとアプリケーション・サーバーが別々のマシンに置かれている場合、ネットワークの通信量が増大しますが、CPU、ディスク領域、利用可能なメモリーなど、システム・リソースが増えるという利点もあります。マルチマシン構成の場合も、インプロセス・サーバーがあると、Oracle9iAS Reports Services コンポーネント間の通信が高速化されるので、パフォーマンスが向上します。

また、Web サーバーとアプリケーション・サーバーをそれぞれ複数のマシンに置くという方法もあります。この方法には追加構成が必要ですが、Web サーバーのロード・バランシングが可能になります。

レポートを複数の言語で配布する場合は、言語ごとに 1 つ以上の Reports Server をセットアップするようお勧めします。

1.4.4 複数サーバーのクラスタ化の選択

クラスタとは、複数のサーバーを 1 つのコミュニティに仮想的にグループ化したものです。このコミュニティの目的は、要求処理をクラスタの各メンバーに効率的に分散することです。以前のバージョンと異なり、Oracle9iAS Reports Services でのクラスタ化は、マスター/スレーブ型ではなく、ピアレベルで行われます。ピアレベルのクラスタ化では、受信要求の分散と処理に対して、クラスタの全メンバーが同等の役割を担います。受信要求は、クラスタ内の特定の Reports Server ではなく、クラスタ全体に対して送信されます。このため、あるメンバーが停止しても、残りのメンバーが要求負荷の管理を続行します。1 台のマシンの障害が原因でシステム全体がダウンする単一の障害箇所はありません。

各クラスタ・メンバー・マシンは、各サーバー・メンバー上でのレポートの実行が可能になるよう、ほぼ同様に構成する必要があります。つまり、構成ファイルの内容はほぼ同じになります。同じ設定にする必要があるかどうかは、ジョブ関連の設定か、マシン関連の設定かで区別します。ジョブ関連の設定は、すべてのクラスタ・メンバーで同じにする必要があります。ジョブ関連の設定には、セキュリティ、データ・ソースおよび宛先タイプに関する設定などがあります。マシン関連の設定には、maxEngine、minEngine、maxIdle、initEngine などの属性があり、これらはメンバーごとに異なる値を設定できます。

また、クラスタ・メンバーは、次の事項に従う必要があります。

- サーバー関連の環境変数には、同じ値を設定します。
- TNS 設定では、同じデータベースを同じ方法で指定します。

同じクラスタのメンバーとなるサーバーは、クラスタ名を共有し（各サーバー名に追加されます）、同じ公開鍵および秘密鍵を設定する必要があります。

ジョブ関連の構成設定が異なるマシンがあると、クラスタ化の利点を享受できません。

多言語対応のために複数のサーバーをセットアップする場合は、言語ごとに 1 つ以上のクラスタをセットアップするようお勧めします。

Oracle9iAS Reports Services の 起動および停止

この章では、Oracle9iAS Reports Services の起動および停止について説明します。この章の主な項は、次のとおりです。

- Reports Server の起動
- Reports Servlet および Reports Server が実行されていることの確認
- Oracle HTTP Server が実行されていることの確認
- Reports Server の停止

注意： この章の各例で使用されている `ORACLE_HOME` は、Oracle9i Application Server がインストールされている場所を示しています。これには、Oracle9iAS Reports Services も含まれます。

Web 上でレポートを実行する場合は、事前に Oracle HTTP Server を起動する必要があります。この操作の詳細は、Oracle9iAS のドキュメントを参照してください。この章の手順はすべて、Oracle HTTP Server が事前に起動されていることを前提としています。

2.1 Reports Server の起動

Reports Server の実行には、次のオプションがあります。

- サービスとしての Reports Server のインストールおよび起動 (Windows NT/2000)
- サブレットとしての Reports Server の起動 (Windows および UNIX)
- コマンドラインからの Reports Server の起動 (Windows および UNIX)

次の各項では、これらのオプションを設定する方法について説明します。

2.1.1 サービスとしての Reports Server のインストールおよび起動 (Windows NT/2000)

Reports Server は、デフォルトでインプロセス・サーバーとしてインストールされますが、Windows NT/2000 マシンでは、必要に応じてサービスとしてインストールすることができます。この作業を行うには、コマンド・プロンプトで次のように入力します。

```
rwserver -install <server_name> [batch=yes/no] [autostart=yes/no]
```

batch のデフォルトは、*no* です。インストール中に確認のためのプロンプトを表示する必要がない場合は、*yes* と入力します。*autostart* のデフォルトは、*no* です。再起動時に、サービスを自動的に開始して、ユーザーがログオンして手動で Reports Server を起動する手間を省く場合は、*yes* と入力します。

このサーバーがクラスタのメンバーである場合は、このコマンドにクラスタ名を追加します。次に例を示します。

```
rwserver -install <server_name>.<cluster_name> [batch=yes/no] [autostart=yes/no]
```

サーバーのクラスタ化の詳細は、第 6 章「Reports Server のクラスタ」を参照してください。

注意： Reports Server NT/2000 サービスを削除するには、コマンド・プロンプトで `rwserver -uninstall <server_name>` と入力します。サーバーがクラスタのメンバーである場合は、クラスタ名を追加します。たとえば、`rwserver -uninstall <server_name>.<cluster_name>` と入力します。

Windows NT/2000 上で Reports Server を起動する手順は、次のとおりです。

1. Reports Server をホスティングするマシンで、「スタート」⇒「設定」⇒「コントロールパネル」を選択し、「コントロールパネル」で「サービス」をダブルクリックします。
2. 「サービス」ダイアログ・ボックスで「Oracle Reports Server [*repserver*]」(*<repserver>* は Reports Server インスタンスの名前)を選択し、「スタートアップ」をクリックします。「サービス」ダイアログ・ウィンドウが表示されます。

3. 「サービス」ダイアログ・ウィンドウで、「ログオン」セクションの「アカウント」をクリックし、オペレーティング・システムのユーザー名とパスワードを選択します。これで、サーバー・プロセスを実行するユーザー・アカウントが指定されます。

注意： PostScript またはプリンタに出力する場合は、Reports Server サービスを実行しているユーザーが、確実にデフォルト・プリンタへのアクセス権限を保持するようにしてください。そのためには、Windows NT/2000 サービスに対してアカウントを設定する際に、プリンタへのアクセス権限を持つ特定のユーザーを指定します。システム・アカウントは、通常、プリンタへのアクセス権限を持っていません。

さらに、Reports Server サービスを実行しているユーザーは、サーバーが必要とする可能性のあるすべてのサービスに対してアクセス権限を持っている必要があります。たとえば、サーバーが別のドライブへの書き込みアクセス権限を必要とする場合があります。

4. 「スタートアップの種類」を「自動」に設定します。
5. 「OK」をクリックします。
6. 「スタート」をクリックします。

「サービス コントロール」メッセージ・ボックスに、Reports Server が起動した日時が表示されます。

2.1.2 サブレットとしての Reports Server の起動（Windows および UNIX）

Reports Server をインプロセス・サーバー（デフォルトの構成）として使用している場合は、要求を送信するだけでサブレットが起動されます。ただし、コマンドラインから要求を送信する場合は、最初に URL からサブレットを起動する必要があります。サブレットの起動に成功した場合は、HTTP Server の起動も成功したことになります。

URL から Reports Servlet を起動するには、Web ブラウザで次のように入力します。

```
http://<your_machine_name>:<your_port_num>/reports/rwervlet
```

2.1.3 コマンドラインからの Reports Server の起動（Windows および UNIX）

Windows NT/2000 では、次のコマンドを使用して、Reports Server をスタンドアロン・サーバーとして起動することもできます。

```
rwserver server=<server_name>
```

サーバーを起動する際に、ダイアログ・ボックスやメッセージを表示しない場合は、コマンド・ライン・キーワード BATCH を追加します。

```
rwserver server=<server_name> batch=yes
```

UNIX では、次の構文を使用してこのコマンドを実行できます。

```
rwserver.sh server=<server_name>
```

または

```
rwserver.sh server=<server_name> batch=yes
```

実行可能ファイルが PATH 環境変数でアクセスできれば、このコマンドは任意のディレクトリから実行できます。

2.2 Reports Servlet および Reports Server が実行されていることの確認

Oracle9iAS Reports Servlet が実行されていることを確認するには、次の URL にナビゲートします。

```
http://<your_machine_name>.<domain_name>:<your_port_number>/reports/rwervlet/help
```

URL は、大文字と小文字を区別することに注意してください。この URL が正常に実行されると、rwervlet コマンドライン引数を説明するヘルプ・ページが表示されます。

Reports Server が実行されていることを確認するには、次の URL にナビゲートします。

```
http://<your_machine_name>.<domain_name>:<your_port_number>/reports/rwervlet/showjobs?server=<server_name>
```

デフォルトの Reports Server 名 (rep_<machine_name>) またはサーブレット構成ファイル rwervlet.properties (%ORACLE_HOME%\reports\conf¥) に指定された Reports Server を使用している場合は、server=<server_name> 引数は必要ありません。この URL が正常に実行されると、指定した Reports Server のジョブ・キュー内のリストが表示されます。

注意： サーブレット構成ファイルの詳細は、[第 3 章「Oracle9iAS Reports Services の構成」](#)を参照してください。

2.3 Oracle HTTP Server が実行されていることの確認

Oracle HTTP Server が実行されていることを確認するには、ブラウザから次の URL にナビゲートします。

```
http://<server_name>.<domain>:<port_number>/
```

2.4 Reports Server の停止

この項では、Windows NT/2000 および UNIX 上で Reports Server を停止する方法について説明します。

- Reports Server が、Windows NT/2000 上でサービスとして実行されている場合は、「サービス」コントロール・パネルから停止します。
- Reports Server が、Windows NT/2000 上でバッチ・ファイルを通じて実行されている場合、または UNIX 上でシェル・スクリプトを通じて実行されている場合は、「Oracle Reports Server」ダイアログ・ボックスの「停止」ボタンをクリックします。
- Oracle Enterprise Manager を起動して、停止する Reports Server までナビゲートし、この Reports Server のホーム・ページで「停止」ボタンをクリックします。Reports および Oracle Enterprise Manager の詳細は、[第 13 章「Oracle9iAS Reports Services の管理と監視」](#)を参照してください。
- Reports Server が Reports Servlet を通じてインプロセス・サーバーとして実行されている場合は、Oracle Program Manager (OPMN) から OC4J を停止します。OPMN の詳細は、Oracle9iAS のドキュメントを参照してください。
- Reports Server が Windows NT/2000 または UNIX 上でコマンドラインから実行されている場合は、コマンド・プロンプトで次のコマンドを入力します。

Windows NT/2000 および UNIX の場合 (UNIX では `rwserver` のかわりに `rwserver.sh` を使用)

次のコマンドで、サーバーが正常に停止します (つまり保留中のジョブを終了した後に停止)。

```
rwserver server=<server> shutdown=normal authid=<admin/pword>
```

次のコマンドで、サーバーが直ちに停止します (つまり保留中のジョブを終了せずに停止)。

```
rwserver server=<server> shutdown=immediate authid=<admin/pword>
```

次のコマンドで、関連メッセージを表示せずにサーバーが停止します。

```
rwserver server=<server> shutdown=normal authid=<admin/pword> batch=yes
```

`rwserver` コマンドで使用するキーワードは、[付録 A「コマンドライン引数」](#)で説明します。

Oracle9iAS Reports Services の構成

Oracle9i Application Server (Oracle9iAS) をインストールすると、Reports Services が自動的に構成されます。環境をカスタマイズするために調整が必要なこともあります。環境全体または大部分を設定する必要はありません。

この章では、主に、カスタマイズを行ったり、デフォルト構成に対する理解を深める際の参考となる情報を提供します。サービス関連の構成ファイルを示し、ほとんどのファイルの内容を詳細に説明します。この章の主な項は、次のとおりです。

- [Oracle9iAS Reports Services 構成ファイル](#)
- [Oracle9iAS Reports Server の構成](#)
- [Reports Servlet の構成](#)
- [URL エンジンの構成](#)

注意： この章の各例で使用されている `ORACLE_HOME` は、Oracle9i Application Server (Oracle9iAS) がインストールされている場所を示しています。Oracle9iAS には、Oracle9iAS Reports Services も含まれます。

構成の際には、環境変数も設定します。環境変数は、インストール時に自動的に設定されません。環境変数の詳細は、[付録 B「Reports 関連の環境変数」](#)を参照してください。

3.1 Oracle9iAS Reports Services 構成ファイル

この項では、Oracle9iAS Reports Services の各種構成ファイルについて説明します。多くの場合、これらのファイルをユーザーが操作する必要はありません。これらのファイルは、サーバー環境の多くの部分をコントロールするので、ユーザーが不適切に変更すると、サーバー環境に不具合が発生する可能性があります。構成ファイルをユーザー自身で変更する場合は、必ず現行バージョンのバックアップを作成してください。

Oracle9iAS Reports Services の構成ファイルは、Reports Server および Reports Servlet に関連します。これらは、[表 3-1](#) で説明します。

注意： [表 3-1](#) で指定されているパスは、Windows を前提として円記号 (¥) を使用していますが、Windows および UNIX どちらの環境でも同じパスです。

表 3-1 Oracle9iAS Reports Services 構成ファイル

コンポーネント	構成ファイル
Reports Server	<p><code>%ORACLE_HOME%\¥reports¥conf¥<server_name>.conf</code></p> <p>この XML ファイルを使用して、Reports キャッシュ、Reports Engine およびセキュリティの初期値の定義、有効な宛先タイプの登録、ログを記録する情報の指定、およびその他のサーバー関連値の設定を行います。</p> <p>このファイルは、サーバー起動時に自動的に作成されます。サーバー名を変更する際、このファイルに入力したカスタム構成設定を保持する場合は、このファイルを新たなサーバー名に変更してからサーバー名を変更します。この順序で行わないと、新しいデフォルト構成ファイルがサーバーによって自動的に作成されます。</p> <p>このファイルの詳細は、「Oracle9iAS Reports Server の構成」を参照してください。</p>
Reports Server	<p><code>%ORACLE_HOME%\¥reports¥dtd¥rwserverconf.dtd</code></p> <p>このファイルには、<server_name>.conf および rwbuilder.conf の要素と属性のデータ型の定義が格納されています。データ型定義では、関連の XML ファイルで使用できるすべての要素、これらの要素に関連する属性、およびこれらの属性のデフォルト値が示されます。</p> <p>このファイルの詳細は、「Reports Server の構成要素 (rwserverconf.dtd)」を参照してください。</p>

表 3-1 Oracle9iAS Reports Services 構成ファイル (続き)

コンポーネント	構成ファイル
Reports Server	<p><code>%ORACLE_HOME%\reports\conf\%server_name%\rwbuilder.conf</code></p> <p>この XML ファイルを使用して、Builder と同じプロセスで実行される Reports Server を構成します。要求はすべて Reports Server 経由で実行される必要があるため、レポートを実行するサーバーが Reports Builder に必要です。Reports Builder は、要求を処理する Reports Server を自動的に起動します。Builder からレポートを実行する際、このファイルは起動するインプロセス・サーバー・インスタンスに構成を提供します。<code><server_name>.conf</code> ファイルと同様に、このファイルのデータ型定義は <code>rwserverconf.dtd</code> ファイルに依存します。ただし、compatible、persistFile および security の要素など、いくつかの要素を除きます。</p> <p>このファイルは <code><server_name>.conf</code> と同じ構成要素を多く含むので、このファイルの構成に必要な情報は、「Oracle9iAS Reports Server の構成」を参照してください。</p>
Reports Servlet	<p><code>%ORACLE_HOME%\reports\conf\%server_name%\rwservlet.properties</code></p> <p>このファイルでは、Reports のキー・マップ・ファイル (<code>cgicmd.dat</code>) の位置とファイル名、および Reports Servlet のインプロセス・サーバーを使用するかどうかなどを指定します。</p> <p>このファイルの詳細は、「Reports Servlet の構成」を参照してください。</p>

3.2 Oracle9iAS Reports Server の構成

Oracle9iAS Reports Services の Reports Server コンポーネントは、`<server_name>.conf` および `rwbuilder.conf` の各 XML ファイルを使用して構成できます。これらのファイルは、Windows および UNIX とともに次のディレクトリにあります。

`%ORACLE_HOME%\reports\conf\%server_name% or rwbuilder>.conf`

これらのファイルは、どちらもデフォルトのサーバー構成値を含む `rwserver.template` ファイル (Windows および UNIX とともに `%ORACLE_HOME%\reports\conf\rwserver.template`) によってサポートされています。

`<server_name>.conf` ファイルは、デフォルトのサーバー構成ファイルです。`rwbuilder.conf` ファイルは、Reports Builder によってインプロセスで使用されるサーバー・インスタンスを構成します。要求はすべて Reports Server 経由で実行される必要があるため、レポートを実行するサーバーが Reports Builder に必要です。Builder からレポートを実行する際、`rwbuilder.conf` ファイルは、Reports Builder と同じプロセスで実行されるサーバー・インスタンス (インプロセス・サーバー) に構成を提供します。`<server_name>.conf` および `rwbuilder.conf` のファイルは、ほぼ同じです。唯一の違いは、`rwbuilder.conf` は [compatible](#)、[persistFile](#) または [security](#) の構成要素 (この項で後述) を使用しませんが、`<server_name>.conf` ではこれらを使用することです。

これらのファイルは、次の条件で自動的に作成されます。

- `<server_name>.conf` ファイルは、サーバーを最初に起動したときに作成されます。このファイルは、`rwserver.template` ファイルをベースにしています。
- `rwbuilder.conf` ファイルは、**Reports Builder** からレポートを最初に実行したときに作成されます。このファイルも、`rwserver.template` ファイルをベースにしています。
- 新しい `<server_name>.conf` ファイルは、サーバー名の変更後、サーバーを次に起動したときに作成されます。新しい構成ファイルは、`rwserver.template` ファイルのデフォルト値をベースにしています。変更前のサーバー名に付随する構成を保持する場合は、`<server_name>.conf` ファイルを新たなサーバー名 (`<new_server_name>.conf`) に変更してから、変更された名前のサーバーを起動します。
- いずれかのファイルを削除すると、削除したファイルはサーバーを次に起動したときに再び作成されます。新しいファイルは、`rwserver.template` ファイルのデフォルト値をベースにしています。

次の項では、次のディレクトリ（Windows および UNIX 共通）にある `rwserverconf.dtd` ファイルを使用して、これらのファイルで許可される構文と値を説明します。

```
%ORACLE_HOME%\reports\%dtd%\rwserverconf.dtd
```

3.2.1 Reports Server の構成要素（`rwserverconf.dtd`）

`rwserverconf.dtd` ファイルで提供される次の要素により、Reports Server が構成されます。

- [server](#)
- [compatible](#)
- [cache](#)
- [engine](#)
- [security](#)
- [destination](#)
- [job](#)
- [notification](#)
- [log](#)
- [jobStatusRepository](#)
- [trace](#)
- [connection](#)

- [queue](#)
- [persistFile](#)
- [identifier](#)
- [pluginParam](#)

これらは XML 要素です。XML は大文字と小文字を区別することに注意してください。

また、これらの任意の要素をサーバー構成ファイル（<server_name>.conf）に追加するとき、rwserverconf.dtd ファイル（Windows および UNIX とともに %ORACLE_HOME%\reports\dtd\ に置かれています）で示された順序に従っていると、使用中の XML エディタでエラー・メッセージが表示されません。この順序に従わなくても構成ファイルは機能しますが、rwserverconf.dtd で指定された大文字と小文字に従わない場合には機能しません。

これらの要素は、関連する属性およびサブ要素とともに、次の各項で説明します。

3.2.1.1 server

例

```
<server>
  [One or more configuration specifications]
</server>
```

必須 / オプション

必須です。構成ファイルに 1 組の *server* 要素の開始タグと終了タグを指定できます。

説明

server 要素は、サーバー構成ファイルのコンテンツ領域を開始および終了します。ファイルの階層に関しては、その他すべての要素は *server* 要素に従属します。

3.2.1.2 compatible

例

```
<compatible version="6i"/>
```

必須 / オプション

オプションです。サーバー構成ファイルに 1 つの *compatible* 要素を指定できます。

説明

compatible 要素を使用して、Reports 6i クライアント (RWCLI60.EXE、RWCGI60.EXE、RWQMU60.EXE、RWRQM60.EXE、RWRQV60.EXE) との下位互換性を確保できます。*compatible* を 6i に設定すると、Reports Server は *rwproxy* という実行可能ファイルを使用して、6i クライアントからの要求をリスニングし、それを 9i サーバーに転送します。

compatible には、*version* という 1 つの属性があります (表 3-2 を参照)。

表 3-2 *compatible* 要素の属性

属性	有効値	説明
version	6i	<i>version</i> を 6i に設定すると、Oracle9iAS で Reports 6i クライアントを実行できます。 <i>version</i> を 6i に設定する場合、6i より前のバージョンも Oracle9iAS で実行できますが、保証された動作ではなく、Oracle ではサポートしていません。

compatible 要素を使用する場合、Reports Server の 6i バージョンと同様、Reports Server に対するエントリも *tnsnames.ora* ファイルに必要です。次に例を示します。

```
testsvr.world = (ADDRESS=
  (PROTOCOL=tcp)
  (HOST=testhost.us.oracle.com)
  (PORT=1949)
)
```

互換性をオフにすれば、この作業は必要ありません。互換性をオフにするには、*compatible* 要素をサーバー構成ファイルから削除します。

注意： *compatible* 要素に 6i が設定され、Reports Server に対する TNS エントリが *tnsnames.ora* ファイルに含まれている場合、サーバーがクラスタのメンバーであれば、クラスタ名を追加する必要があります。クラスタ名を使用する場合、*sqlnet.ora* ファイルで指定されたデフォルト・ドメインは除外します。次に例を示します。

```
myserver.world (standalone server with default domain)
myserver.cluster1 (server part of cluster1)
myserver.cluster1.world (invalid entry)
```

3.2.1.3 cache

例

```
<cache class="oracle.reports.cache.RWCache">
  <property name="cacheSize" value="50"/>
  <property name="cacheDir" value="D:\orawin\reports\server\cache"/>
</cache>
```

必須 / オプション

オプションです。サーバー構成ファイルに 1 つの *cache* 要素を指定できます。cache 要素が指定されていない場合は、デフォルトが使用されます (oracle.reports.cache.RWCache)。

説明

cache 要素を使用して、サーバーのキャッシュ実装を定義する Java クラスを指定できます。デフォルトの *cache* Java クラスを使用するか、Oracle9iAS Reports Services Cache API を使用して独自の実装を定義できます。

注意： Reports API の最新情報は、Oracle Technology Network (<http://otn.oracle.com/> 英語サイト) または Oracle Technology Network Japan (<http://otn.oracle.co.jp/> 日本語サイト) を参照してください。

cache 要素には、*class* という 1 つの属性があります (表 3-3 を参照)。

表 3-3 cache 要素の属性

属性	有効値	説明
class	「説明」を参照してください。	デフォルト: oracle.reports.cache.RWCache oracle.reports.cache.Cache インタフェースを実装する完全修飾された Java クラス。

また、任意数のプロパティを *cache* 要素に入力できます。プロパティは、*cache* に登録する実装クラスによって認識および解釈される名前と値のペアです。たとえば、Reports Services で提供されるデフォルトの *cache* Java クラスを使用する場合、構成エントリは次のようになります。

```
<cache class="oracle.reports.cache.RWCache">
  <property name="cacheSize" value="50"/>
  <property name="cacheDir" value="D:\orawin\reports\server\cache"/>
</cache>
```

この例では、*cacheSize* の単位はメガバイトで、キャッシュの位置を示す *cacheDir* は Windows プラットフォーム向けに指定されています。UNIX では、次のような UNIX 標準を使用します。

```
<property name="cacheDir" value="/ORACLE_HOME/reports/server/cache"/>
```

cache を実装する際に追加情報へのアクセスが必要な場合は、その情報を *pluginParam* 要素で指定します。

3.2.1.4 engine

例

```
<engine id="rwEng" class="oracle.reports.engine.EngineImpl" initEngine="1"
maxEngine="5" minEngine="1" engLife="50" maxIdle="15" callbackTimeOut="60000">
  <property name="sourceDir" value="D:\orawin\reports\myReport"/>
  <property name="tempDir" value="D:\orawin\reports\myTemp"/>
</engine>
```

必須 / オプション

必須です。サーバー構成ファイルに 1 つ以上の *engine* 要素を指定する必要があります。

説明

engine 要素は、完全修飾された Java クラスを識別します。この Java クラスは、エンジンを起動し、エンジンの動作制御を設定する多数の属性を提供します。Reports Services で提供されるデフォルトのエンジンを使用するか (`oracle.reports.engine.EngineImpl`)、Oracle9iAS Reports Services Engine API を使用して独自の実装を定義できます。カスタム・エンジンの例としては、データベース・イベントの発生時に特定のオペレーティング・システム・コマンドを実行するエンジンなどを開発できます。

注意： Reports API の最新情報は、Oracle Technology Network (<http://otn.oracle.com/> 英語サイト) または Oracle Technology Network Japan (<http://otn.oracle.co.jp/> 日本語サイト) を参照してください。

engine には、複数の属性があります (表 3-4 を参照)。

表 3-4 engine 要素の属性

属性	有効値	説明
id	文字列	特定の <i>engine</i> 要素を識別する、構成 XML ファイル内で一意のキーワード。テキスト文字列または数字のどちらでも使用できます。次に例を示します。 <code>id="rwEng"</code>
class	「説明」を参照してください。	デフォルト: <code>oracle.reports.engine.EngineImpl</code> <code>oracle.reports.engine.Engine</code> および <code>oracle.reports.engine.EngineInterface</code> の 2 つのインタフェースを実装する完全修飾された Java クラス。
classPath	文字列	<i>class</i> 属性で指定される Java クラスへのディレクトリ・パス。ディレクトリは、サーバー・プラットフォームの表記規則に従って指定します。次に例を示します。 Windows: <code>classPath="%ORACLE_HOME%\myEngine.jar"</code> UNIX: <code>classPath="\$ORACLE_HOME/myEngine.jar"</code>
initEngine	数値	デフォルト: 1 初期化時に Reports Server で起動するエンジンの数。
maxEngine	数値	デフォルト: 1 サーバーで実行できるこのタイプのエンジンの最大数。
minEngine	数値	デフォルト: 0 サーバーで維持されるこのタイプのエンジンの最小数。
engLife	数値	デフォルト: 50 エンジンが終了する前、および必要に応じて新しいエンジンが開始する前に、エンジンで実行できるジョブの数。この機能により、メモリー・リークを防止できます。

表 3-4 engine 要素の属性（続き）

属性	有効値	説明
maxIdle	数値	<p>デフォルト : 30</p> <p>エンジンの現在数が <i>minEngine</i> を超えている場合に、エンジンが停止する前に許可されるアイドル時間（分単位）。</p> <p>たとえば <i>minEngine</i> が 0、<i>maxIdle</i> が 30 であり、1 つのエンジンが実行中の場合、このエンジンは 30 分間使用されないと停止します。同じ条件で <i>minEngine</i> が 1 の場合、30 分間アイドル状態であっても実行中のエンジンは停止しません。</p>
callbackTimeOut	数値	<p>デフォルト : 60000</p> <p>サーバーがエンジンを呼び出してからエンジンがサーバーに応答するまでに許可される待ち時間（ミリ秒）。</p> <p>サーバーをホスティングするマシンが非常に高速である場合、小さい数を指定するとパフォーマンスが向上します。</p>

また、任意数のプロパティを *engine* 要素に入力できます。唯一の要件は、**Reports** エンジンを実装する Java クラスによって識別される名前と値のペアであることです。たとえば、**Reports Services** で提供されるデフォルトの *engine* Java クラスを使用する場合、*engine* の構成エントリは次のようになります。

```
<engine id="rwEng" class="oracle.reports.engine.EngineImpl" initEngine="1"
maxEngine="5" minEngine="1" engLife="50" maxIdle="15" callbackTimeOut="60000">
  <property name="sourceDir" value="D:\orawin\reports\myReport"/>
  <property name="tempDir" value="D:\orawin\reports\myTemp"/>
</engine>
```

この例で、*sourceDir* と *tempDir* は Windows 環境用の設定です（UNIX では、*sourceDir*="\$ORACLE_HOME/reports/myReport" と *tempDir*="\$ORACLE_HOME/reports/myTemp" になります）。*sourceDir* プロパティは、レポート定義ファイルに使用するデフォルト・ディレクトリを識別します。このプロパティは、**REPORTS_PATH** 環境変数で指定したパス情報を上書きします。

tempDir プロパティは、**Reports Services** でテンポラリ・ファイルに使用するテンポラリ・ディレクトリの名前と位置を識別します。この値がデフォルト・エンジンに対して指定されていない場合は、**REPORTS_TMP** 環境変数で指定されるテンポラリ・ディレクトリが使用されます。**REPORTS_TMP** も指定されていない場合は、オペレーティング・システムのデフォルト・テンポラリ・ディレクトリが使用されます。

この構成ではデフォルトのエンジン・クラスが使用されるので、*classPath* 属性は指定されません。

エンジンが、送信メール・サーバーなど追加情報へのアクセスを必要とする場合は、その情報を *pluginParam* 要素で指定します。

3.2.1.5 security

例

```
<security id="rwSec" class="oracle.reports.server.RWSecurity">
  <property name="securityUserId" value="portal_id/portal_password@portal_schema"
    confidential="yes" encrypted="no"/>
</security>
```

必須 / オプション

オプションです。*security* 要素を構成ファイルに入力しないと、Reports Server のセキュリティが確保されません。任意数の *security* 要素を構成ファイルに入力できます。

説明

security 要素は、サーバー・アクセスを制御する完全修飾された Java クラスを識別します。Reports Services で提供されるデフォルトの *security* クラスを使用するか（このクラスは、Oracle9iAS に含まれる Oracle9iAS Portal 経由で提供されるセキュリティ機能に依存します）、Reports Server Security API を使用して独自の実装を定義できます。

注意： Reports API の最新情報は、Oracle Technology Network (<http://otn.oracle.com/> 英語サイト) または Oracle Technology Network Japan (<http://otn.oracle.co.jp/> 日本語サイト) を参照してください。

security の属性は、表 3-5 を参照してください。

表 3-5 security 要素の属性

属性	有効値	説明
id	文字列	特定の <i>security</i> 要素を識別する、構成 XML ファイル内で一意のキーワード。テキスト文字列または数字のどちらでも使用できます。次に例を示します。 id="rwSec"

表 3-5 security 要素の属性（続き）

属性	有効値	説明
class	「説明」を参照してください。	デフォルト : oracle.reports.server.RWSecurity Reports Server Security Java インタフェース (oracle.reports.server.Security) を実装する完全修飾された Java クラス。デフォルトは、Oracle9iAS に含まれる Oracle9iAS Portal 経由で提供されるセキュリティ機能に依存します。

また、複数のプロパティを security 要素に入力することもできます。唯一の要件は、Reports Server のセキュリティを実装する Java クラスによって識別される名前と値のペアであることです。たとえば、Reports Services で提供されるデフォルトの security Java クラスを使用する場合、security の構成エントリは次のようになります。

```
<security id="rwSec" class="oracle.reports.server.RWSecurity">
  <property name="securityUserid" value="portal_id/portal_password@portal_schema"
    confidential="yes" encrypted="no"/>
</security>
```

この例では、接続情報が提供され、Reports Server で Oracle9iAS Portal のセキュリティ機能にアクセスできるようになります。confidential および encrypted のプロパティ属性は、プロパティ内の情報の暗号化に使用されます。confidential="yes" および encrypted="no" の属性を入力すると、サーバー再起動後、プロパティ値が自動的に暗号化されます。構成ファイルを次にオープンする際、パスワード情報が暗号化され、encrypted が yes に設定されます。構成ファイルに入力したパスワードを忘れた場合は、プロパティを削除して新しい値を再入力し、encrypted を no に設定します。

security を実装する際に追加情報へのアクセスが必要な場合は、その情報を [pluginParam](#) 要素で指定します。

クラスタ環境でセキュリティを設定する場合、予期しない動作を防ぐため、各クラスタ・メンバーに同じセキュリティ・ポリシーを使用する必要があります。

3.2.1.6 destination

例

```
<destination destype="oraclePortal" class="oracle.reports.server.DesOraclePortal">
  <property name="portalUserid" value="portal_id/portal_password@portal_schema"
    confidential="yes" encrypted="no"/>
</destination>
```

必須 / オプション

オプションです。destination 要素をサーバー構成ファイルに入力しないと、提供されている宛先クラスが使用されます（プリンタ、メール、ファイル、キャッシュ、および Oracle9iAS Portal。ポータルへのログインに使用されるユーザー ID とパスワードを指定するには、サーバー構成ファイルにエントリが必要なので、ポータルは例外です）。任意数の destination 要素をサーバー構成ファイルに入力できます。

説明

destination 要素を使用して、サーバーとともに宛先タイプを登録できます。Oracle9iAS Portal を除き、プリンタ、電子メール、ファイルまたはキャッシュなど、提供されている（デフォルト）宛先の登録は不要です。Oracle9iAS Reports Services Destinations API で作成した宛先タイプは登録する必要があります。

注意： Reports API の最新情報は、Oracle Technology Network (<http://otn.oracle.com/> 英語サイト) または Oracle Technology Network Japan (<http://otn.oracle.co.jp/> 日本語サイト) を参照してください。

宛先の構成は、第 4 章「Oracle9iAS Reports Services の宛先の構成」を参照してください。

destination の属性は、表 3-6 で説明します。

表 3-6 destination 要素の属性

属性	有効値	説明
destype	文字列	宛先タイプを識別します。次に例を示します。 destype="printer"
class	「説明」を参照してください。	Reports Server Destination Java クラス (oracle.reports.server.Destination) のサブクラスである完全修飾された Java クラス。使用可能な値は、次のとおりです。 ■ oracle.reports.server.DesMail ■ oracle.reports.server.DesFile ■ oracle.reports.server.DesPrinter ■ oracle.reports.server.DesOraclePortal

また、複数のプロパティを destination 要素に入力することもできます。唯一の要件は、Reports Server Destination Java クラスのサブクラスである Java クラスによって識別される名前と値のペアであることです。次に例を示します。

```
<destination destype="oraclePortal" class="oracle.reports.server.DesOraclePortal">
  <property name="portalUserid" value="portal_id/portal_password@portal_schema"
    confidential="yes" encrypted="no"/>
</destination>
```

この例では、接続情報が提供され、Reports Server で Oracle9iAS Portal にアクセスできるようになります。*confidential* および *encrypted* の属性が含まれているので、Reports Server の次回起動時に *portalUserid* 値が自動的に暗号化されます。

destination を実装する際に追加情報が必要な場合は、その情報を *pluginParam* 要素で指定します。

3.2.1.7 job

例

```
<job jobType="report" engineId="rwEng" securityId="rwSec"/>
```

必須 / オプション

必須です。1 つ以上の *job* 要素を指定する必要があります。

説明

job 要素は、*engine* および *security* の要素と連携して動作します。*job* を使用して、ジョブ・タイプを識別し、そのタイプのジョブで使用するエンジンおよびセキュリティの実装を指定します。たとえば、ユーザーが開発した、データベース・イベントの発生時に特定のオペレーティング・システム・コマンドを実行するエンジンなどです。Oracle9iAS Report Service のイベントドリブンの公開 API を使用して、イベントを特定のジョブ・タイプとして識別します。そのイベントが発生したとき、ジョブ・タイプ情報が Reports Server に渡され、Reports Server が構成ファイルの *job* 要素でそのジョブ・タイプを検索し、要素の属性で提供される指示に従います。この指示は、そのタイプのジョブに指定されたエンジン（および場合によってはセキュリティの実装）に対して指定されています。

job 要素の属性は、表 3-7 で説明します。

表 3-7 job 要素の属性

属性	有効値	説明
jobType	文字列	デフォルト : report サーバーで処理されるジョブのタイプを記述します。Reports Server に備わっているエンジンがそのジョブ・タイプを処理できれば、任意のジョブ・タイプを入力できます。

表 3-7 job 要素の属性 (続き)

属性	有効値	説明
engineId	ID 参照	このジョブ・タイプを処理するエンジンに対して入力された ID を参照します。利用可能な ID は、サーバー構成ファイルで <i>engine</i> 要素の <i>id</i> 属性を使用して指定されています。 <i>id</i> は、特定のエンジンを識別する、構成 XML ファイル内で一意のキーワードです。
securityId	ID 参照	このジョブ・タイプに適用されるセキュリティ・メカニズムに対して入力された ID を参照します。利用可能な ID は、サーバー構成ファイルの <i>security</i> 要素で指定されています。

3.2.1.8 notification

例

```
<notification id="tellMe02" class="oracle.reports.server.MailNotify"/>
```

必須 / オプション

オプションです。*notification* 要素を構成ファイルに入力しないと、通知機能は使用できません。任意数の *notification* 要素を構成ファイルに入力できます。

説明

notification 要素を使用して、ジョブが成功または失敗したときに送られる通知のタイプを定義する Java クラスを指定します。電子メール経由で通知を送信するデフォルトの *notification* クラスを使用するか、Reports Notification API を使用して独自のクラスを定義できます。

注意： Reports API の最新情報は、Oracle Technology Network (<http://otn.oracle.com/> 英語サイト) または Oracle Technology Network Japan (<http://otn.oracle.co.jp/> 日本語サイト) を参照してください。

notification 要素の属性は、表 3-8 で説明します。

表 3-8 notification 要素の属性

属性	有効値	説明
id	文字列	デフォルト : mailNotify 特定の <i>notification</i> 要素を識別する、構成 XML ファイル内で一意のキーワード。テキスト文字列または数字のどちらでも使用できます。次に例を示します。 id="tellMe01"
class	ID 参照	デフォルト : oracle.reports.server.MailNotify Reports Server Notification Java クラス (oracle.reports.server.Notification) を実装する完全修飾された Java クラス。

デフォルトの電子メールによる通知実装を使用する場合は、*pluginParam* 要素を使用して、メールの送信に使用する送信用 SMTP メール・サーバーを指定します。ランタイム・コマンド *notifysuccess* および *notifyfailure* を使用して、通知の送信先となる電子メール・アドレスを指定します（詳細は、付録 A「コマンドライン引数」を参照してください）。たとえば、これらのコマンドを次のようにランタイム URL に挿入できます。

```
notifysuccess=<recipient's e-mail address>&notifyfailure=<recipient's e-mail address>
```

デフォルトの電子メール実装では、*notification* のタイプごとに 1 つのアドレスのみを指定できます。一方または両方の通知タイプを指定できます。通知は、同じアドレスまたはそれぞれ異なるアドレスに送信できます。

サーバー構成ファイルの *notification* 要素は、次のようになります。

```
<notification id="mailNotify" class="oracle.reports.server.MailNotify"/>  
  <property name="succNoteFile" value="succnote.txt"/>  
  <property name="failNoteFile" value="failnote.txt"/>
```

デフォルトの通知実装では、成功または失敗のテキスト・ファイルがデフォルト位置 (%ORACLE_HOME%\reports\templates) にある限り、パスの指定は不要です。デフォルト位置にない場合は、サーバーをホスティングするプラットフォームの要件に従ってディレクトリ・パスとファイル名を入力します。

3.2.1.9 log

例

```
<log option="allJobs"/>
```


必須 / オプション

オプションです。サーバー構成ファイルに 1 つの *log* 要素を指定できます。

説明

log 要素は、下位互換性を確保するために用意されています。この要素は、**Reports** ログ・ファイルの生成および書き込みを行います。ログ・ファイルは自動的に生成され、次のパスに格納されます（このパスは、Windows と UNIX で共通です）。

```
%ORACLE_HOME%\reports\logs\*.log
```

log 要素には、*option* という 1 つの属性があります（表 3-9 を参照）。

表 3-9 log 要素の属性

属性	有効値	説明
option	allJobs succeededJobs failedJobs noJob	デフォルト: noJob デフォルトでログを記録するサーバー・アクティビティの他に、ログを記録するジョブのタイプを記述します。次のオプションがあります。 <ul style="list-style-type: none">■ allJobs: すべてのジョブのログを記録します。■ succeededJobs: 正常に実行されたジョブのログのみを記録します。■ failedJobs: 失敗したジョブのログのみを記録します。■ noJob: ジョブのログを記録しません。

3.2.1.10 jobStatusRepository

例

```
<jobStatusRepository class="oracle.reports.server.JobRepositoryDB">  
  <property name="repositoryConn" value="scott/tiger@ORCL" confidential="yes"  
    encrypted="no"/>  
</jobStatusRepository>
```

必須 / オプション

オプションです。サーバー構成ファイルに 1 つの *jobStatusRepository* 要素を指定できます。

説明

jobStatusRepository 要素は、ジョブ・ステータス・リポジトリを実装する Java クラスを指定します。この要素は、*persistFile* 要素に加えて、ジョブ・ステータス情報を格納するためのもう 1 つの手段を提供します。

persistFile はバイナリ・ファイルであるため、アプリケーション内でジョブ・ステータス情報を公開する際には使用できません。*jobStatusRepository* は、ステータス情報を格納する追加手段を提供することで、アプリケーションのステータス情報を保存する手段を提供します。

デフォルトのクラス `oracle.reports.server.JobRepositoryDB` では、情報はデータベースに格納されます。Reports API を使用して、目的の位置に情報を格納するよう、Reports Server Job Repository インタフェース (`oracle.reports.server.JobRepository`) を独自に実装できます。

注意：

Reports API の最新情報は、Oracle Technology Network (<http://otn.oracle.com/> 英語サイト) または Oracle Technology Network Japan (<http://otn.oracle.co.jp/> 日本語サイト) を参照してください。

jobStatusRepository 要素には、*class* という 1 つの属性があります (表 3-10 を参照)。

表 3-10 *jobStatusRepository* 要素の属性

属性	有効値	説明
class	「説明」を参照してください。	デフォルト: <code>oracle.reports.server.JobRepositoryDB</code> Reports Server Job Repository Java クラス (<code>oracle.reports.server.JobRepository</code>) を実装する完全修飾された Java クラス。

jobStatusRepository 要素では、任意数のプロパティを使用して、引数をリポジトリに渡すことができます。唯一の要件は、ユーザーが指定した名前と値のペアを、サーバー構成ファイルで指定するクラスが認識することです。

サーバー構成ファイルの *jobStatusRepository* 要素は、次のようになります。

```
<jobStatusRepository class="oracle.reports.server.JobRepositoryDB">  
  <property name="repositoryConn" value="scott/tiger@ORCL" confidential="yes"  
    encrypted="no"/>  
</jobStatusRepository>
```

この例で、*repositoryConn* プロパティの値は、リポジトリが格納されているデータベースにアクセスするためのログイン情報です。*confidential* および *encrypted* の属性が使用されているので、Reports Server の再起動時にログイン情報が暗号化されます。

ジョブ・ステータス・リポジトリに追加情報が必要な場合は、その情報を *pluginParam* 要素で指定します。

3.2.1.11 trace

例

```
<trace traceFile="neptune.trc" traceOpts="trace_prf|trace_dbg|trace_wrn"
traceMode="trace_append"/>
```

必須 / オプション

オプションです。サーバー構成ファイルに 1 つの *trace* 要素を指定できます。

説明

trace 要素を使用して、レポート実行のトレースに使用するファイルを作成し、トレースするオブジェクトとアクティビティを指定します。*trace* 要素は、サーバーとエンジンのみのトレースをコントロールします。

注意： サーブレットと JSP のトレースは、サーブレット構成ファイル (rwsvrlet.properties) で構成します (3.3 項を参照)。個別レポートのトレースは、レポートのランタイム・コマンドラインで指定できます (付録 A「コマンドライン引数」を参照)。

trace の属性は、表 3-11 で説明します。

表 3-11 trace 要素の属性

属性	有効値	説明
traceFile	*.trc	デフォルト : <server_name>.trc トレース・ファイルの名前。パスの指定がない場合、トレース・ファイルは、Windows と UNIX の両方で次のディレクトリに置かれます。 ORACLE_HOME/reports/logs/
traceOpts	表 3-12 を参照してください。	デフォルト : trace_all この属性は、トレースするアクティビティを定義します。1 つ以上の traceOpt 値を指定できます。次に例を示します。 <traceOpts="trace_prf trace_brk"> 値は縦線 () で区切ります。有効値は、表 3-12 を参照してください。

表 3-11 trace 要素の属性（続き）

属性	有効値	説明
traceMode	trace_replace	デフォルト : trace_replace
	trace_append	新しいトレース情報で既存のトレース・ファイルを上書きするか（trace_replace）、既存のトレース情報を残したままトレースの末尾に追加するか（trace_append）を決定します。

表 3-12 traceOpts 属性の有効値

値	説明
trace_prf	サーバーおよびエンジン・プロファイルのログを記録します。
trace_brk	デバッグ・ブレークポイントを示します。
trace_app	すべてのレポート・オブジェクトに関する情報を記録します。
trace_pls	すべての PL/SQL オブジェクトに関する情報を記録します。
trace_sql	すべての SQL に関する情報を記録します。
trace_tms	トレース・ファイルの各エントリのタイムスタンプを入力します。
trace_dst	配布リストを示します。 どのレポート・セクションがどの宛先に送信されたかを判断できます。
trace_log	トレース・ファイルのログ情報を複製します。 サーバー構成ファイルで <i>trace</i> 要素の他に <i>log</i> 要素も指定した場合、ログ・ファイルに出力された情報がトレース・ファイルにも送信されます。
trace_err	サーバーのエラー・メッセージを示します。
trace_inf	他のオプションでトレース・ファイルに記録されない情報をすべて記録します。
trace_dbg	デバッグ情報を記録します。
trace_wrn	サーバーの警告メッセージを示します。
trace_sta	初期化中、準備完了、実行中および停止など、サーバーおよびエンジンの状態に関する情報を提供します。
trace_all	サーバーおよびエンジンに関するすべての情報をトレース・ファイルに記録します。

複数のトレース要素を指定するときは、縦線で区切ります。次に例を示します。
traceOpts="trace_prf|trace_dbg|trace_wrn"

3.2.1.12 connection

例

```
<connection maxConnect="50" idleTimeOut="20">
  <orbClient id="RWClient" publicKeyFile="clientpub.key"/>
  <cluster publicKeyFile="%ORACLE_HOME%\reports\server\serverpub.key"
    privateKeyFile="%ORACLE_HOME%\reports\server\serverpri.key"/>
</connection>
```

必須 / オプション

オプションです。サーバー構成ファイルで *connection* 要素を指定しない場合は、デフォルト値が使用されます (表 3-13 を参照)。サーバー構成ファイルに 1 つの *connection* 要素を指定できます。

説明

connection 要素は、サーバーとサーバーに接続されているクライアント間の接続ルールを定義します。

connection の属性は、表 3-13 で説明します。

表 3-13 connection 要素の属性

属性	有効値	説明
maxConnect	数値	デフォルト: 20 サーバーが同時にサービスを提供できる要求の最大数。要求数が <i>maxConnect</i> 値を超えると、Java 例外が返されます。
idleTimeOut	数値	デフォルト: 15 接続のアイドル状態が許可される時間 (分)。

connection には、*orbClient* と *cluster* という 2 つのサブ要素があります。

orbClient は、クライアントが Reports Server との接続に使用する公開鍵ファイルの名前を提供します。任意数の *orbClient* サブ要素をサーバー構成ファイルに入力できます。

orbClient 要素の属性は、表 3-14 で説明します。

表 3-14 orbClient 要素の属性

属性	有効値	説明
id	文字列	デフォルト : RWClient 公開鍵および秘密鍵を使用するレポート・クライアントを識別します。 また、Pluggable clients API でカスタム・クライアントを指定できます。Oracle9iAS Reports Services API の詳細は、Oracle Technology Network (http://otn.oracle.com/ 英語サイト) または Oracle Technology Network Japan (http://otn.oracle.co.jp/ 日本語サイト) を参照してください。
publicKeyFile	<filename>.key	デフォルト : clientpub.key クライアントが Reports Server との接続に使用する公開鍵ファイルの名前を識別します。デフォルト・ファイルは、rwrun.jar ファイルに格納されます。

デフォルトで提供されているクライアントの公開鍵ファイルおよび秘密鍵ファイルは、clientpub.key および clientpri.key です。これらの鍵ファイルは、Oracle9iAS Reports Services のすべてのコンポーネントで使用されます。公開鍵ファイルおよび秘密鍵ファイルを再生成し、デフォルトの鍵ペアを置換できます。この作業を行うには、コマンド・プロンプトで次のように入力します。

```
java oracle.reports.utility.KeyManager <path_and_client_public_key_file_name> <path_and_client_private_key_file_name>
```

これらの鍵を再生成する場合、公開鍵ファイルの位置を publicKeyFile 属性で指定して、rwrun.jar ファイル (%ORACLE_HOME%\reports\jlib\rwrun.jar) の秘密鍵ファイルを置換できます。このためには、jar 形式のファイルを元の形式に戻し、再生成した秘密鍵を入力した後、ファイルを再び jar 形式に変換します。

cluster サブ要素は、すべてのクラスタ・メンバーが使用する公開鍵ファイルおよび秘密鍵ファイルを指定します。サーバー構成ファイルには、cluster 要素を 1 つ指定するか、または指定しません。

注意： サーバー・クラスタの詳細は、第 6 章「Reports Server のクラスタ」を参照してください。

同じクラスタのメンバーとなるサーバーは、同じ拡張クラスタ名と公開鍵および秘密鍵を持つ必要があります。つまり、拡張クラスタ名 (serverA.clus、serverB.clus の場合、.clus が拡張クラスタ名) が等しく、公開鍵ファイルおよび秘密鍵ファイルが各クラスタ・メンバーのサーバー構成ファイル (<server_name>.conf) で指定されている必要があります。

デフォルトのサーバーの公開鍵および秘密鍵は、Windows と UNIX の両方で、パス %ORACLE_HOME%\reports\jlib\rwrun.jar にある rwrun.jar ファイルに格納されています。ただし、クラスタで使用するよう新たに生成した公開鍵および秘密鍵を jar 形式に変換する必要はありません。これらの鍵は任意の場所に置き、それらの絶対パスとファイル名をサーバー構成ファイルで指定します。

3.2.1.13 queue

例

```
<queue maxQueueSize="1000"/>
```

必須 / オプション

オプションです。サーバー構成ファイルに 1 つの queue 要素を指定できます。queue 要素を指定しない場合、デフォルトの 1000 が使用されます。

説明

queue 要素は、Reports の各キューで保持できるジョブの最大数を指定します。Reports Services には、次の 3 つのキュー・コンポーネントがあります。

- スケジュールされたジョブのキュー
- 実行中のジョブのキュー
- 完了したジョブのキュー

queue 要素は、これらの各コンポーネントに許可される値を指定します。ジョブ数がキューに指定された最大値を超えると、期限切れのジョブ、最も古いジョブの順に自動的に削除されます。また Reports Queue Manager を使用して、キューに保持されているジョブの数を手動で減らすこともできます。

注意： 詳細は、Reports Queue Manager のオンライン・ヘルプを参照してください。

queue 要素には、maxQueueSize という 1 つの属性があります (表 3-15 を参照)。

表 3-15 queue 要素の属性

属性	有効値	説明
maxQueueSize	数値	デフォルト: 1000 Reports のジョブ・キューに保持できるジョブの最大数。

3.2.1.14 persistFile

例

```
<persistFile filename="neptune.dat"/>
```

必須 / オプション

オプションです。ファイルを指定しない場合、サーバーは `<server_name>.dat` というデフォルト名で独自のファイルを作成します。1 つの *persistFile* 要素を指定できます。

説明

persistFile 要素は、すべてのジョブ・ステータスを記録するファイルを識別します。Reports Server は、このファイルを使用してサーバーを停止前のステータスに復元します。

このファイルの名前 *persistFile* は、サーバーが停止後に再起動しても元の状態のままであること、つまり永続性があることに由来します。

サーバー永続ファイルは、サーバーの初回起動時、または現在のサーバー永続ファイルを削除または名前変更した後の初回起動時に自動的に作成されます。このファイル名を変更した後も引き続き使用する場合は、新しい名前をサーバー構成ファイルに入力してから、実際にファイル名を変更し、次にサーバーを再起動します。

persistFile 要素には、*fileName* という 1 つの属性があります (表 3-16 を参照)。

表 3-16 persistFile 要素の属性

属性	有効値	説明
fileName	文字列	デフォルト :<server_name>.dat サーバー永続ファイルの名前、およびオプションでパスを指定します。ファイルが次のデフォルト・ディレクトリにある場合は、パスを省略できます。 %ORACLE_HOME%\reports\server\ このパスは、Windows と UNIX の両方で共通です。

3.2.1.15 identifier

例

```
<identifier confidential="yes"  
encrypted="yes">fpoiVNFvnlkjRPortn+sneU88=NnN</identifier>
```

必須 / オプション

オプションです。サーバー構成ファイルに 1 つの *identifier* 要素を指定できます。

説明

identifier 要素は、管理者として Reports Queue Manager に最初にログインしたときに、構成ファイルに自動的に書き込まれます。初回ログイン時、Queue Manager の管理者ユーザー ID とパスワードが設定されます。この情報は暗号化された後サーバー構成ファイルに書き込まれ、その後すべての Queue Manager のログイン情報の認証に使用されます。

Queue Manager のログイン情報を忘れた場合、この情報をサーバー構成ファイルから削除して Reports Server 構成ファイルに次のフォーマットで再入力します。

```
<identifier confidential="yes" encrypted="no">username/password</identifier>
```

Queue Manager の次回起動時に、ID が自動的に暗号化され、encrypted が yes にリセットされます。次のようになります。

```
<identifier confidential="yes"
encrypted="yes">fpoiVNFvnlkjRPortn+sneU88=NnN</identifier>
```

Reports Queue Manager の詳細は、Reports Queue Manager のオンライン・ヘルプを参照してください。

3.2.1.16 pluginParam**例**

```
<pluginParam name="mailServer">smtp01.mycorp.com</pluginParam>
```

必須 / オプション

オプションです。pluginParam 要素は、いくつでも指定できます。

説明

pluginParam 要素は、Oracle9iAS Reports Services のすべてのトランスポート・コンポーネントと連携して動作します。トランスポート・コンポーネントには、エンジン、セキュリティ、キャッシュ、宛先、および jobstatusRepository があります。これらには、メール・サーバー、FTP URL またはその他のタイプのプラグインへのアクセスが必要な場合もあります。pluginParam 要素は、すべてのトランスポート・コンポーネントで使用できるプラグインを指定する手段を提供します。この要素を使用すると、この情報をトランスポート・コンポーネントのクラス定義に含める必要がなく、プラグインのソースをすばやく簡単に変更できます。

たとえば、カスタムのトランスポート・エンジンおよび宛先 Java クラスの両方にプロキシ・サーバーへのアクセスが必要であるとします。これら両方のクラスでプロキシ・サーバーへのアクセスをハードコーディングするかわりに、proxy など必要なプラグイン・タイプを呼び出して、その位置をサーバー構成ファイルの pluginParam で示せばよいだけです。

トランスポートابل・コンポーネントでサポートおよび必要とされるプラグインであり、トランスポートابل・コンポーネントが認識する名前であれば、どのようなプラグインでも指定でき、また自由に命名できます。

pluginParam の属性は、表 3-17 で説明します。

表 3-17 *pluginParam* 要素の属性

属性	有効値	説明
name	文字列	プラグイン・パラメータの名前。
type	text	デフォルト : text
	file	指定されているプラグインのタイプ。
	url	<ul style="list-style-type: none">■ <i>text</i> の場合は、メール・サーバーの名前など、指定されているプラグインを識別できる文字列を指定します。<i>text</i> は、<i>pluginParam</i> 要素の内容がテキストであることを示すので、<i>getPluginParam()</i> メソッドは要素に指定されている内容をそのまま返します。■ <i>file</i> の場合は、プラグイン・ファイルのディレクトリ・パスとファイル名を指定します。ディレクトリ・パスは、Reports Server のホスト・マシン (Windows または UNIX) の標準に従って指定します。<i>file</i> は、<i>pluginParam</i> 要素の内容がファイル名であることを示すので、<i>getPluginParam()</i> メソッドは指定されたファイルから読み込んだ内容を返します。■ <i>url</i> の場合は、FTP サイトへの完全な URL など、プラグインで必要とする完全な絶対 URL を指定します。URL は、<i>pluginParam</i> 要素の内容が URL であることを示すので、<i>getPluginParam()</i> メソッドはその URL から読み込んだ内容を返します。使用する URL と Oracle9iAS Reports Services は、ファイアウォールの同じ側にある必要があります。
	デフォルト・タイプ (text) を使用するときは、 <i>pluginParam</i> 文字列で指定する必要はありません。この項の最初に示した例では、プラグイン (メール・サーバー名) がデフォルト・タイプの text であるので、タイプの指定がありません。	

3.3 Reports Servlet の構成

次のパス（Windows と UNIX で共通）にある `rwervlet.properties` というファイルを使用して、Reports Servlet を構成します。

```
%ORACLE_HOME%\reports\conf\rwervlet.properties
```

サーブレットでは、CGI 実装を使用していたユーザーにはなじみのあるコンポーネントが使用されます。

Reports Servlet 構成ファイルを使用して、次の処理を行います。

- キー・マップ・ファイルの位置の指定
- キー・マップ・ファイルの再ロード
- 冗長エラー・メッセージの非表示
- ログイン・ダイアログ・ボックスの選択
- Reports Servlet と JSP のトレース・オプションの設定
- サーバー・エラー・メッセージの外観のカスタマイズ
- インプロセス・サーバーの指定
- デフォルトの Reports Server の識別
- 動的に生成されたイメージの指定
- DB 認証と SYSAUTH Cookie の期限切れの設定
- DB 認証 Cookie 用の暗号化キーの設定
- 診断 / デバッグ出力のフォーマットの追加
- SSL ポート番号の指定
- `rwervlet` ヘルプ・ファイルの定義
- シングル・サインオン使用の指定

構成ファイルのエントリでは、大文字と小文字を区別しません。

Windows の場合、サーブレット構成ファイルでは、1 つの円記号ではなく 2 つの円記号 (%%) を使用してディレクトリ・パスを指定します。最初のスラッシュは次のスラッシュをエスケープします。エスケープしない円記号は、このファイルで別の意味になります。たとえば、Windows の Reports Servlet ファイルで、次のパスはその下のパスのようにになります。

```
d:\orawin\reports\conf\filename.ext
```

次のようになります。

```
d:%%orawin%%reports%%conf%%filename.ext
```

UNIX の場合、ディレクトリ・パスは、次のようにプラットフォームの標準に従って指定します。

```
orawin/reports/conf/filename.ext
```

3.3.1 キー・マップ・ファイルの位置の指定

レポートのランタイム・コマンドラインには、ユーザーには公開しない情報が含まれることもあります。また、オプションが長い文字列で構成されていて覚えにくかったり、読みづらい URL になることもあります。

レポートのコマンドライン引数をキー・マップ・ファイル (cgicmd.dat) に入力して、必要な引数が格納されているこのファイル内の特定のキー・セクションの名前だけをランタイム・コマンドとして公開することができます。

キー・マップ・ファイルの詳細は、[第 8 章「レポート要求の実行」](#)を参照してください。Reports Servlet 構成ファイルを使用して、このファイルの位置を示します。

次に例を示します。

```
KeyMapFile=d:¥¥orawin¥¥reports¥¥conf¥¥cgicmd.dat
```

この例では、デフォルトのファイル名と位置を使用しています。サーブレットはすでに検索対象と検索位置を認識しているので、デフォルトではキー・マップ・ファイルの位置とファイル名に対するエントリがサーブレット構成ファイルに表示されません。別の名前または位置のファイルを使用する場合は、ディレクトリ・パスとファイル名を格納する KeyMapFile パラメータをサーブレット構成ファイルで指定する必要があります。

3.3.2 キー・マップ・ファイルの再ロード

RELOAD_KEYMAP パラメータを使用して、サーブレットが要求を受け取るたびにキー・マップ・ファイル (cgicmd.dat) を再ロードするかどうかを指定します。

次に例を示します。

```
RELOAD_KEYMAP=yes
```

この機能は、マップ・ファイルを頻繁に変更し、変更を自動的にロードする場合に便利です。ファイルの再ロードに要する時間は、ランタイム・パフォーマンスに影響します。

一般にこのパラメータは、本番環境では *no* に設定し、テスト環境では *yes* に設定します。

3.3.3 冗長エラー・メッセージの非表示

ランタイム・エラー発生時に生成されるエラー・メッセージの本文に、ランタイム・コマンドラインの詳細を表示できます。ただし、ランタイム・コマンドラインにユーザー ID やパスワードなど機密情報が含まれている場合、エラー・メッセージのテキストとともにそれらを表示するのは好ましくありません。

サブレット構成ファイルの **DIAGNOSTIC** パラメータを使用して、コマンドライン情報をそのエラー・メッセージとともに表示するかどうかを指定できます。

コマンドラインの詳細を表示しない場合は、**DIAGNOSTIC** パラメータに **NO** を指定します。これがデフォルト値です。次に例を示します。

```
DIAGNOSTIC=NO
```

すべてのエラー・メッセージに詳細なランタイム・コマンドラインを表示する場合は、**YES** を指定します。

3.3.4 ログイン・ダイアログ・ボックスの選択

サブレット構成ファイルは、ユーザー ID およびパスワードのダイアログ・ボックスのテンプレートを処理する多数のパラメータを提供します。これらのダイアログ・ボックスは、ユーザーがデータベースにログインしたときやセキュア・レポートを実行したときにオープンされます。一般に、これらのパラメータはログイン画面の設定に使用される各種テンプレートを示します。これらのテンプレートは、自社ロゴ、リンク・ボタン、およびその他の HTML を自由に使用してカスタマイズできます。

DBAUTH および **SYSAUTH** のパラメータを使用して、個別のログイン画面に使用する HTML テンプレートの位置とファイル名を指定できます。

たとえば、次のエントリは、データベース・ログイン画面のテンプレートを示します。

```
DBAUTH=dbauth.htm
```

SYSAUTH は、セキュア・レポートのログイン画面を示します。次に例を示します。

```
SYSAUTH=sysauth.htm
```

テンプレートが次に示すデフォルトのテンプレート・ディレクトリにある場合は、テンプレートのパスを指定する必要はありません。

```
%ORACLE_HOME%\reports\templates
```

DB_SYS_DIFFAUTH および **DB_SYS_SAMEAUTH** のテンプレートでは、これらのログイン画面が 1 つの画面に統合されます。**DB_SYS_DIFFAUTH** は、ユーザーがデータベースとレポートへのアクセスでそれぞれ異なるログイン情報の入力が必要な場合に使用します。

次に例を示します。

```
DB_SYS_DIFFAUTH=dbsysdif.htm
```

DB_SYS_SAMEAUTH テンプレートは、ユーザーが両方のタイプのアクセスで同じログイン情報を入力する場合に使用します。

次に例を示します。

```
DB_SYS_SAMEAUTH=dbsyssam.htm
```

この場合、ユーザーはログイン情報の入力を 1 回だけ要求され、入力値はサーブレットによってデータベースと Reports Server の両方に渡されます。

3.3.5 Reports Servlet と JSP のトレース・オプションの設定

トレースは、Reports Services の各種コンポーネントについて多数の異なるタイプのランタイム情報を記録できるよう、Reports Services 環境にあらかじめ追加されています。

注意： Reports Server のトレースは、サーバー構成ファイル (<server_name>.conf) で構成します。詳細は、[3.2.1.11 項](#)を参照してください。個別レポートのトレースは、レポートのランタイム・コマンドラインで指定できます ([付録 A「コマンドライン引数」](#)を参照)。

Reports Servlet および JSP のランタイム情報を追跡および記録するには、サーブレット構成ファイルの TRACEOPTS パラメータを使用します。また、任意数のトレース・オプションを入力できます。各オプションは、縦線で区切ります。

次に例を示します。

```
TRACEOPTS=trace_prf|trace_pls|trace_dbg
```

使用可能なすべてのトレース・オプションは、[表 3-12](#) で説明します。

また、TRACEFILE および TRACEMODE のパラメータを使用できます。

TRACEFILE を使用して、トレース・ファイルの名前を指定できます。次に例を示します。

```
TRACEFILE=<server_name>.trc
```

パスの指定がない場合、トレース・ファイルは、Windows と UNIX の両方で次のディレクトリに置かれます。

```
%ORACLE_HOME%\reports\logs
```

TRACEMODE を使用して、新しいトレース情報で既存のトレース・ファイルを上書きするか (trace_replace)、既存のトレース情報を残したままトレースの末尾に追加するか (trace_append) を決定します。次に例を示します。

```
TRACEMODE=trace_append
```

TRACEMODE のデフォルトは、trace_replace です。

3.3.6 サーバー・エラー・メッセージの外観のカスタマイズ

Reports Services は、サーバー・エラー・メッセージのテンプレートを提供します。これらのメッセージは、原因に従って自動的に生成されます。テンプレートは、エラー・メッセージの表示に使用される視覚的な設定を提供します。

エラー・メッセージの外観は、自社ロゴやエラーと関連するアイコンなどを使用してカスタマイズできます。ヘルプ・システムや自社ホームページを表示したり、前のブラウザ・ウィンドウへ戻るためのボタンを追加することもできます。これには、自動的に生成されたエラー・メッセージに独自の HTML フレームワークを設定します。

エラー・メッセージ・テンプレートの名前と位置を示すエントリをサーブレット構成ファイルに入力します。

デフォルトのエントリは次のとおりです。

```
ERRORTEMPLATE=rwerror.htm
```

テンプレートが次に示すデフォルトのテンプレート・ディレクトリにある場合は、エラー・メッセージ・テンプレートのパスを指定する必要はありません。

```
%ORACLE_HOME%\reports\templates
```

3.3.7 インプロセス・サーバーの指定

Reports Server を Reports Servlet と同じプロセスで実行することを選択した場合、SERVER_IN_PROCESS パラメータを適切に設定します。Reports Server をインプロセス・サーバーとして実行するには、サーブレット構成ファイルで次のように指定します。

```
SERVER_IN_PROCESS=yes
```

Reports Server を Reports Servlet と同じプロセスで実行しない場合は、no を入力します。

注意： インプロセス・サーバーの実行の長所と短所は、[第 1 章「Oracle9iAS Reports Services のアーキテクチャ」](#)を参照してください。

3.3.8 デフォルトの Reports Server の識別

Reports Servlet は、SERVER パラメータを使用してデフォルトの Reports Server を識別します。ランタイム URL などサーバー名が指定されない場合は、ここで指定したデフォルトのサーバーが使用されます。デフォルトの Reports Server 名をサーブレット構成ファイルに入力します。

次に例を示します。

```
SERVER=<server_name>
```

デフォルトの Reports Server がサーバー・クラスタのメンバーである場合は、次のようにクラスタ名を使用します。

```
SERVER=<cluster_name>
```

サーバー名とクラスタ名の組合せを使用した場合、デフォルトのサーバーに送信された要求はこの特定のマシンに渡されます。このマシンが停止した場合、エラー・メッセージが返され、レポートは実行されません。クラスタ名のみを指定すると、要求は任意のクラスタ・メンバーに送信され、送信先のマシンにアイドル状態のエンジンがない場合は、別のマシンに転送されます。

rwervlet.properties で SERVER パラメータを指定しない場合は、デフォルトのサーバー名は rep_<machine_name> です。

3.3.9 動的に生成されたイメージの指定

IMAGE_URL パラメータを使用して、動的に生成された Reports イメージにアクセスできる位置を指定できます。

次に例を示します。

```
IMAGE_URL=http://<server_or_web_server_name>.<domain_name>:<port>/reports/rwervlet
```

このパラメータは、Reports Servlet 経由で実行されない JSP で使用されます。これにより、チャートなど動的に生成されたイメージを表示できるのが、レポート実行者に制限されます。Reports Servlet 経由で実行される JSP やその他のレポート・タイプには、この保護が自動的に付加されます。

3.3.10 DB 認証と SYSAUTH Cookie の期限切れの設定

COOKIEEXPIRE パラメータを使用して、データベースおよびシステム認証 Cookie の存続期間を分単位で指定できます。次に例を示します。

```
COOKIEEXPIRE=20
```

デフォルトは 30 です。

Cookie は、ユーザーの初回認証時に、暗号化されたユーザー名とパスワードをクライアント側で保存します。サーバーは、クライアントから Cookie を受け取ると、Cookie に保存された時間と現在のシステム時間を比較します。この時間が COOKIEEXPIRE に定義された時間（分）を超えていると、サーバーはその Cookie を拒否し、エラー・メッセージとともにデータベース認証フォームをクライアントに返します。レポートを実行するには、ユーザーは再度認証を受ける必要があります。

3.3.11 DB 認証 Cookie 用の暗号化キーの設定

ENCRYPTIONKEY を使用して、DB 認証 Cookie のユーザー名とパスワードを暗号化するための暗号化キーを指定できます。暗号化キーには任意の文字列を設定できます。次に例を示します。

```
ENCRYPTIONKEY=egbdf
```

3.3.12 診断 / デバッグ出力のフォーマットの追加

DIAGBODYTAGS および DIAGHEADTAGS のパラメータを使用して、追加の HTML エンコードを、診断およびデバッグの出力に関連する出力ファイルの <body> および <head> のタグに挿入できます。

DIAGBODYTAGS は <body> タグ全体を定義し、DIAGHEADTAGS は開始および終了の <head>/</head> タグの間に使用できるタグを定義します。

これらのパラメータを使用して、診断およびデバッグの出力を見やすくするためのフォーマット引数を挿入できます。次に例を示します。

```
DIAGBODYTAGS=<BODY [additional HTML encoding]>
```

```
DIAGHEADTAGS=<HEAD>[additional HTML encoding]</HEAD>
```

3.3.13 SSL ポート番号の指定

Secure Sockets Layer (SSL) を使用するときデフォルト (443) 以外のポートを使用する場合は、SSLPORT を使用します。次に例を示します。

```
SSLPORT=<port number>
```

3.3.14 rwservlet ヘルプ・ファイルの定義

HELP キーワードを rwservlet コマンドとともに使用して、サーブレット関連のヘルプ・トピックを表示できます。ヘルプ・ファイルを呼び出すには、次の URL を指定します。

```
http://yourwebserver/yourervletpath/rwservlet/help
```

注意： HELP キーワードの詳細は、[付録 A「コマンドライン引数」](#)を参照してください。

このパラメータを指定しない場合に表示されるサーブレット用のデフォルトのヘルプ・ファイルもあります（デフォルトの情報は、コードから取得され、ファイルではありません）。ユーザー独自のヘルプ・ファイルを追加することもできます。これには、サーブレットのヘルプ・ファイルの名前と位置の URL を、HELPURL パラメータでサーブレット構成ファイルに指定します。次に例を示します。

```
HELPURL=http://your_web_server/your_help_file_path/helpfile.htm
```

3.3.15 シングル・サインオン使用の指定

Oracle9iAS Reports Services のシングル・サインオン機能を利用するには、サーブレット構成ファイルで SINGLESIGNON パラメータを使用して指定する必要があります。シングル・サインオンを使用してユーザー認証を行う場合は YES と入力し、シングル・サインオンを使用しない場合は NO と入力します。NO を選択した場合、Reports Server は独自の認証メカニズムを使用してユーザー認証を行います（Reports6i で使用されているものと同じ認証メカニズム）。

シングル・サインオンを使用してユーザー認証を行うには、次のように入力します。

```
SINGLESIGNON=yes
```

3.4 URL エンジンの構成

Reports Server は、URL のコンテンツの取得および配布を行う URL エンジンを搭載しています。URL エンジンを使用すると、Reports Server の強力なスケジューリングおよび配布の機能を利用して、任意の公開 URL のコンテンツを電子メール、Oracle9iAS Portal および WebDav などの宛先に配布できます。Reports Server の宛先はトランスポートブルなので、URL コンテンツに対するユーザー独自のカスタム宛先を追加することもできます。

さらに、URL エンジンを Reports Server の Event-Driven Publishing API と組み合わせて使用すれば、コンテンツの配布をデータベース・イベントでトリガーできます。たとえば、従業員経費データのテーブルに格納されたデータを高品位の Web で公開するための JSP レポートを作成したとします。この場合、URL エンジンと Event-Driven Publishing API を使用して、新規または更新した従業員経費データが経費アプリケーションによってテーブルに格納されるたびに、その JSP に電子メールで通知することができます。

URL エンジンとは、Oracle9iAS Reports Services のインストール時にデフォルトで有効化されません。URL エンジンを実効にする手順は、次のとおりです。

1. URL エンジンの **engine** 要素をサーバー構成ファイルに追加します。たとえば、次のようにエンジン要素を入力します。

```
<engine id="rwURLEng"
      class="oracle.reports.engine.URLEngineImpl"
      initEngine="1"
      maxEngine="1"
      minEngine="0"
      engLife="50"
      maxIdle="30"
      callbackTimeOut="60000"
/>
```

2. 適切なジョブ・タイプを URL エンジンと関連付ける **job** 要素をサーバー構成ファイルに追加します。たとえば、次のようにジョブ要素を入力します。

```
<job  jobType="urlEngine"
      engineId="rwURLEng"
/>
```

3. Reports Server を停止し、再起動します。

注意： これらの新しい要素を含む Reports Server を再起動するとき、Reports Server のステータス・メッセージ・ボックスでエンジン数の増加を確認できます。前述の例では、Reports Server を再起動するとエンジンの数が 1 つ増加します (initEngine の値)。

URL エンジンへの要求送信の詳細は、第 8 章「レポート要求の実行」を参照してください。

3.5 プロキシ情報の入力

Oracle9iAS Reports Services のいくつかの機能は、ファイアウォール経由での情報の送受信をサポートします。たとえば、URL エンジン、XML データ・ソース、テキスト・データ・ソースおよびメール宛先の機能は、すべてファイアウォール経由で情報を送受信します。これらの機能が適切に動作するために、Reports Server は特定のプロキシ情報を必要とします。処理を簡素化するために、必要なプロキシ情報を一個所に格納し、Reports Server 構成ファイルからそれを指定するようにします。Reports Server にプロキシ情報を構成する手順は、次のとおりです。

1. **pluginParam** 要素をサーバー構成ファイルに追加し、プロキシ情報ファイル (proxyinfo.xml など) を指定するようにします。たとえば、次のように **pluginParam** 要素を入力します。

```
<pluginParam name="proxy" type="file">proxyinfo.xml</pluginParam>
```

注意： オプションでプロキシ情報ファイルのパスを指定できます。このファイルのデフォルトの位置は、<ORACLE_HOME>/reports/conf です。

2. 構成に必要なプロキシ値でプロキシ情報ファイルを更新します。たとえば、次のように proxyinfo.xml に入力します。

```
<proxyInfo>
  <proxyServers>
    <proxyServer name="xyz.abc.com" port="80" protocol="http"/>
    <proxyServer name="www-proxy1.xyz.abc.com" port="80" protocol="ftp"/>
    <proxyServer name="www-prox21.xyz.abc.com" port="80" protocol="https"/>
  </proxyServers>
  <bypassProxy>
    <domain>*.abc.com</domain>
  </bypassProxy>
</proxyInfo>
```

注意： デフォルトのプロキシ情報ファイル<ORACLE_HOME>/reports/conf/proxyinfo.xml を参照してください。

3.6 Oracle Enterprise Manager 用の Reports Server の構成

Oracle9iAS に搭載されている Oracle Enterprise Manager (OEM) は、Oracle9iAS Reports Services に管理サービスおよび監視サービスを提供します。これらのサービスを利用するには、Reports Server と連携して動作するよう OEM を構成する必要があります。Oracle9iAS のインストール時に構成および起動されるデフォルトの Reports Server インスタンスでは、OEM との連携が自動的に構成されます。別の Reports Server インスタンスを追加する場合は、次のコマンドラインを実行して OEM との連携を構成する必要があります。

UNIX の場合：

```
$ORACLE_HOME/bin/addNewServerTarget.sh <reports_server_name>
```

Windows の場合：

```
%ORACLE_HOME%\bin\addNewServerTarget.bat <reports_server_name>
```

注意： デフォルトで、このスクリプトは、Oracle9iAS Portal インストールのユーザー名とパスワードを、OEM と Reports Server の統合でのユーザー名とパスワードとして使用します。Oracle9iAS Portal がインストールされていない場合、デフォルトのユーザー名は repadmin、パスワードは reports です。

Reports Server に対する OEM の使用方法の詳細は、[第 13 章「Oracle9iAS Reports Services の管理と監視」](#)を参照してください。

Oracle9iAS Reports Services の宛先の構成

レポートを実行する際には、レポートの出力方法（宛先）と受信者（配布）を検討する必要があります。配布については、[第 9 章「拡張配布の作成」](#)で説明しています。この章では、Oracle9iAS Reports Services でデフォルトの宛先やカスタムの宛先に対し出力が処理される方法について説明します。まず出力処理の概要を示し、次に Oracle9iAS Reports Server への宛先タイプの登録について説明します。

この章には、次の項があります。

- [出力処理の概要](#)
- [サーバーへの宛先タイプの登録](#)

4.1 出力処理の概要

レポートの出力方法は、実行時に指定する宛先タイプによって制御されます。指定可能な宛先タイプは、各自のサーバー構成ファイル (<server_name>.conf) に登録した宛先出力タイプによって決まります。ユーザーは出力タイプを登録できず、Oracle9iAS Reports Services で提供する次のデフォルト・タイプを使用します。

- キャッシュ（つまりブラウザ）
- SMTP 対応電子メール
- ファイル
- プリンタ
- Oracle9iAS Portal（ポータルへアクセスするには、サーバー構成ファイルにユーザー ID とパスワードを指定する必要があるため、このタイプは例外です）

Oracle9iAS Reports Services Destinations API を使用して、FAX、FTP、Oracle の Internet File System（iFS）などのカスタム出力タイプや、必要な任意のタイプを定義することもできます。この API を使用して、新しい宛先タイプを定義できます。また、レポートをカスタムの宛先に導くハンドラを作成することもできます。

注意： カスタムの宛先タイプは、Oracle9iAS Reports Services Destinations API で作成します。Reports API とダウンロード用宛先タイプの最新情報は、Oracle Technology Network (<http://otn.oracle.com/> 英語サイト) または Oracle Technology Network Japan (<http://otn.oracle.co.jp/> 日本語サイト) を参照してください。

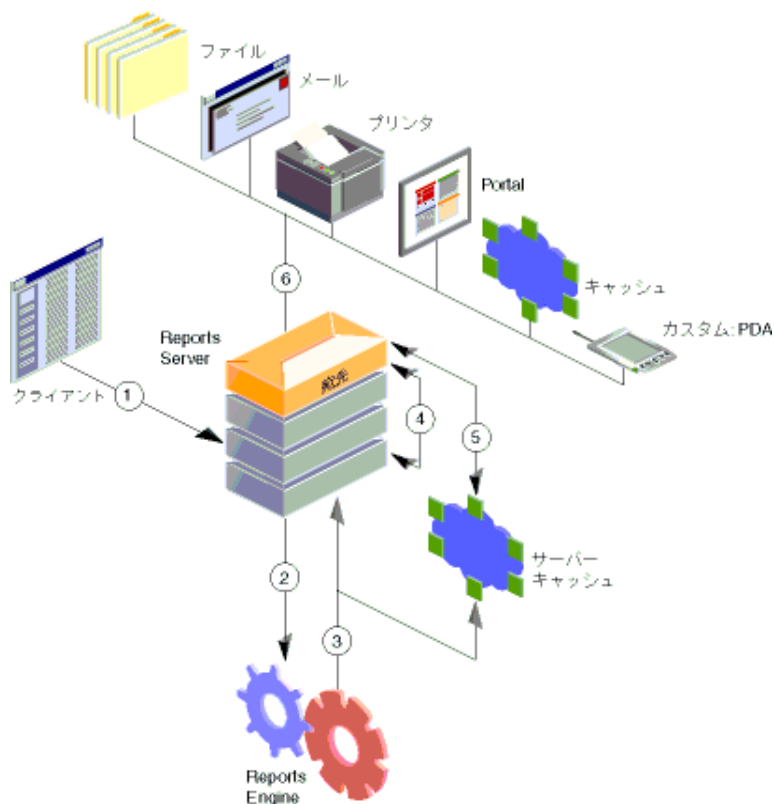
Oracle9iAS Reports Services のアーキテクチャによって、出力の生成と配布の方法が標準化されます。このアーキテクチャでは、Reports Server を通じてレポート出力が適切な宛先に配信され、さらに Reports Engine を通じて出力がその宛先とは関係なく生成されます。これによって、1 回のレポート実行が様々な用途で使用できるので、効率が大幅に向上します。また、出力処理アーキテクチャが拡大され、任意数の宛先タイプに対応できるようになっています。

以前は、Reports ランタイム・エンジンが出力の配信を全面的に処理していました。このため、出力先との通信方法をランタイム・エンジンで認識して必要がありました。この結果、エンジンとサポートされている宛先は密接に結び付いていました。

Oracle9iAS Reports Services は、この密接な結び付きとそれに伴う制限を解消します。ランタイム・エンジンは、すべての宛先を同等に処理するようになりました。ランタイム・エンジンが、作成される出力の宛先タイプを認識する必要はありません。サーバーが出力を宛先ハンドラに渡し、さらにハンドラがそれぞれに関連付けられた宛先タイプに配信するためのデータを準備します。事前定義の宛先タイプ（事前定義ハンドラを使用）を使用するか、またはサポートするカスタムの宛先タイプ用のハンドラを作成することができます。Reports では、ほとんどすべてのタイプの宛先に接続することができます。

図 4-1 は、Oracle9iAS Reports Services の出力処理アーキテクチャの主要なコンポーネントを示しています。

図 4-1 宛先 / 配布アーキテクチャの主要コンポーネント



出力処理アーキテクチャでは、要求が次の順序で処理されます。

1. ユーザーが、クライアントまたはブラウザから Reports Server に要求を送信します。
2. サーバーがこの要求をランタイム・エンジンに渡します。
3. ランタイム・エンジンが、宛先オブジェクト（特定の宛先のファイル・リストとそれらの宛先に関連するすべてのプロパティを含んだもの）およびレポート出力を作成 / 処理したうえで、宛先オブジェクトを Reports Server に、レポート出力をキャッシュにそれぞれ送信します。
4. Reports Server が、宛先オブジェクトを Reports Server の宛先コンポーネントに送信します。

5. Reports Server の宛先コンポーネントが、キャッシュからレポート出力をフェッチします。
6. Reports Server の宛先コンポーネントが、レポートと宛先オブジェクト（宛先デバイスによる出力の処理方法を指定するもの）を適切な宛先ハンドラに送信します。

4.2 サーバーへの宛先タイプの登録

Oracle9iAS Reports Server で特定の宛先タイプにレポートを送信するには、そのタイプがデフォルト・タイプ（プリンタ、電子メール、キャッシュまたはファイル）、または事前にサーバーの構成ファイル（<server_name>.conf）に登録されたタイプであることが必要です。構成ファイルには、レポートに有効な宛先タイプを登録するための *destination* 要素が含まれています。任意数の宛先タイプを任意の場所で登録できます。

サーバーへの宛先タイプの登録には、次のタスクが関係します。

- [サーバー構成ファイルの destination セクションの設定](#)
- [宛先として有効な値の入力](#)

これらのタスクについては、次の各項で説明します。

4.2.1 サーバー構成ファイルの destination セクションの設定

<server_name>.conf ファイルの destination セクションを設定する手順は、次のとおりです。

1. サーバー構成ファイルを任意のテキスト・エディタでオープンします。

サーバー構成ファイルは、次のディレクトリにあります（Windows と UNIX で共通のパス）。

```
%ORACLE_HOME%\reports\conf\<server_name>.conf
```

2. 構成ファイルに *destination* セクションがない場合は、構成ファイルのデータ型定義ファイル（rwserverconf.dtd）セクションにおいて destination セクションの前にある要素の下に、destination セクションを作成します。

注意： サーバー構成ファイルは、関連する文書タイプ定義ファイル（ORACLE_HOME\reports\dtd\rwserverconf.dtd）に定義されている要素の順序に従います。この定義ファイルで宛先の前にある要素がサーバー構成ファイルに存在する場合は、それらの要素の後に *destination* を配置します。

3. レポートを出力する際に使用する宛先タイプを登録するには、常に次の構文を使用します。

```
<destination destype="output_type_1" class="java_class_1">
  <property name="valid_destype_property" value="valid_value"/>
  <property name="valid_destype_property" value="valid_value"/>
</destination>
<destination destype="output_type_2" class="java_class_2">
  <property name="valid_destype_property" value="valid_value"/>
</destination>
```

これらのタグの有効値は、次の項で説明します。

4.2.2 宛先として有効な値の入力

4.2.2.1 宛先タイプとクラス

destype 属性と *class* 属性は、デフォルト以外の出力タイプの有効な登録に対して必要となります。この 2 つの属性で、宛先タイプとそれらに関連する Java クラスを指定します。Oracle9iAS Reports Services に付属の事前定義（デフォルト）の宛先タイプおよびクラスを表 4-1 に示します。

表 4-1 標準の宛先タイプとクラス

宛先	宛先タイプ	クラス
Oracle9iAS Portal コンテンツ領域	oraclePortal	oracle.reports.server.DesOraclePortal
SMTP 対応電子メール	mail	oracle.reports.server.DesMail
ファイル	file	oracle.reports.server.DesFile
キャッシュ	cache	oracle.reports.server.DesCache
プリンタ	printer	oracle.reports.server.DesPrint

デフォルト・タイプのうち、**oraclePortal** 宛先タイプだけは登録する必要があります。これは、**oraclePortal** 宛先タイプがポータルにアクセスするためのユーザー ID とパスワードの指定を必要とするためです。

使用できる宛先タイプとクラスは、サーバーで提供する事前定義済のものに限られません。Destinations API を使用してカスタム・ハンドラを定義すれば、FAX や FTP などのカスタムの宛先タイプを登録できます。

注意： Reports API の最新情報は、Oracle Technology Network (<http://otn.oracle.com/> 英語サイト) または Oracle Technology Network Japan (<http://otn.oracle.co.jp/> 日本語サイト) を参照してください。

4.2.2.2 宛先プロパティの名前 / 値のペア

サーバー構成ファイルでは、登録された宛先に任意数のプロパティを関連付けることができます。宛先のプロパティは名前 / 値のペアで構成され、それぞれのペアが出力タイプの構成のある側面を定義します。これらは、宛先タイプで認識できる用語で表されます。たとえば、宛先タイプが `oraclePortal` である宛先は、次の名前 / 値のペアを認識します。

```
<property name="portalUserId" value="portal_id/portal_password@portal_schema"
confidential="yes" encrypted="no"/>
```

この例では、ポータルユーザー ID に関連付ける値を定義しており、*confidential* および *encrypted* という属性が指定されています。*confidential="yes"* は、この要素内で値の暗号化が必要であることを示し、*encrypted="no"* は値がまだ暗号化されていないことを示しています。次に Reports Server を起動したときは、値が自動的に暗号化され、*encrypted* が *yes* にリセットされます。

注意： サーバー構成ファイルに指定できる要素と属性は、`rwserverconf.dtd` ファイル (`%ORACLE_HOME%\¥reports¥dtd¥rwserverconf.dtd`) に定義されている構文によって決まります。詳細は、第 3 章「[Oracle9iAS Reports Services の構成](#)」を参照してください。

宛先タイプのプロパティに有効な値は、その宛先タイプに完全に依存します。これらの値は、Reports から取得されるのではなく、Reports Server で使用するわけでもありません。これらは宛先タイプ自体から取得され、宛先で認識する用語を使用します。開発者は、カスタムの宛先の要件を理解し、特定のカスタム出力タイプに関連付けるプロパティを把握する必要があります。

配布の検討に入ると、配布 XML ファイル内の *destype* 要素にもプロパティの名前 / 値のペアが指定できることに気付きます。サーバー構成ファイルの *destination* 要素に入力されるプロパティと、配布 XML ファイルの *destype* 要素に入力されるプロパティを区別することが重要です。

- サーバー構成ファイルの *destination* 要素に入力されるプロパティは、出力タイプの設定のみに関係します。たとえば、宛先 FAX の再試行許容回数を設定します。
- 配布 XML ファイルの *destype* 要素に入力されるプロパティは、ランタイム・パラメータの指定のみに関係します。たとえば、FAX の受信者の識別子を指定します。

ユーザー・アクセスの制御

インターネットはあまりにオープンであるために、企業のどの機密情報に誰がアクセスできるかを制御することが課題となります。Oracle9iAS Portal は、Oracle9iAS Reports Services に数多くのセキュリティ機能を提供しています。これを利用して、重要なデータに適切なユーザーが安全にアクセスできるようになります。Oracle9iAS Portal のセキュリティ機能を正しく使うことで、ユーザーにはそれぞれに許可されたデータのみが表示されます。

Oracle9iAS Reports Services を使用して、次の制御を行います。

- 各レポートにアクセスできるユーザー
- レポートを実行できる期間
- レポートの実行に使用できるサーバーとプリンタ
- ユーザーが編集できるレポート・パラメータとその値の範囲

この章では、Oracle9iAS Reports Services のセキュリティ、および Oracle9iAS Portal で提供するデフォルトのセキュリティ実装を使用して、Reports 環境へのユーザー・アクセスを制御する方法について説明します。この章には、次の項があります。

- [Oracle9iAS Portal の概要](#)
- [サーバー構成ファイルにおけるポータルベースのセキュリティの定義](#)
- [Oracle9iAS Portal での Reports ユーザーおよび名前付きグループの作成](#)
- [アクセス制御の設定](#)

セキュリティ制御を設定するには、事前に Oracle9iAS Portal と Oracle9iAS Reports Services の両方がインストールされ、構成されていることが必要です。Reports Services の構成の詳細は、[第3章「Oracle9iAS Reports Services の構成」](#)を参照してください。Oracle9iAS Portal の構成の詳細は、Oracle9iAS Portal のドキュメントを参照してください。また、両コンポーネントのインストールの詳細は、『Oracle9i Application Server インストレーション・ガイド』を参照してください。Oracle9iAS および Oracle9iAS Portal の詳細は、Oracle9iAS のドキュメントが収録された CD および Oracle Technology Network (<http://otn.oracle.com/> 英語サイト) または Oracle Technology Network Japan (<http://otn.oracle.co.jp/> 日本語サイト) を参照してください。

5.1 Oracle9iAS Portal の概要

Oracle9iAS Portal は、データの公開と開発に関するブラウザ・ベースのソリューションで、情報を Web に公開し、Web ベースのデータ駆動型アプリケーションを作成するための Web ベースのツールを提供します。

Oracle9iAS Portal は Oracle9iAS Reports Services と緊密に統合されているため、データを公開するための強力かつ安全な環境を整えることができます。Oracle9iAS Portal には、Oracle9iAS Reports Services のセキュリティを簡単に設定するためのウィザードが用意されています。これには、レポート、Oracle9iAS Reports Server、プリンタ、出力フォーマットおよびレポート・パラメータのユーザー・アクセスをそれぞれを定義するためのウィザードが含まれます。

定義したアクセス制御情報は、Oracle9iAS Portal のリポジトリに格納されます。その後、Oracle9iAS Portal ユーザーとして、登録済の RDF および JSP を Oracle9iAS Portal ページにオプションで公開できます。Oracle9iAS Portal の機能を全面的に使用していても、Portal を使ってレポートを配布する必要はありません。レポートはコマンドラインから配布でき、さらに Oracle9iAS Portal を通じて常にアクセス制御機能も利用できます。

Oracle9iAS Reports Services のセキュリティ機能に対するアクセスは、レポート・リンクまたはレポート内容の公開に Portal を使用しているかどうかに関係しません。Portal を通じて公開しない場合でも、Oracle9iAS Portal で提供されている Oracle9iAS Reports Services のセキュリティ機能を利用して、すべてのレポートに対するユーザー・アクセスを制御できます。

Oracle9iAS Portal を通じてレポートをポートレットとして公開する場合は、Oracle9iAS Reports Services で Oracle9iAS Single Sign-On 機能が使用されます。Oracle9iAS Single Sign-On によって、ユーザーは何度もログインする必要がなくなります。つまり、最初にポータルにログインした後に、ポータル内でポートレットとして公開されている各アプリケーションにログインする必要がありません。Single Sign-On でログインすると、Oracle9iAS Portal によって、登録されたポートレット・プロバイダとサブシステムのすべてに自動的にログインされます。

Single Sign-On の詳細は、『Oracle9i Application Server セキュリティ・ガイド』を参照してください。

5.2 サーバー構成ファイルにおけるポータルベースのセキュリティの定義

セキュリティ機能を使用する場合は、Reports Server 構成ファイル (<server_name>.conf) の security 要素を設定します。このファイルは、UNIX と Windows のどちらのプラットフォームの場合も、次のディレクトリ・パスにあります。

```
%ORACLE_HOME%\reports\conf\<server_name>.conf
```

Portal のデフォルトのセキュリティ実装を使用する場合は、Reports Server 構成ファイルの *security* 要素に、Portal の有効なユーザー名、パスワードおよびデータベース接続文字列 SID を含んだプロパティを指定する必要があります。Reports Server は、この情報を使用してサーバーを Portal に接続し、Portal で設定されたアクセス制御パラメータを取得します。サーバーから Portal への接続はバックグラウンドで実行されるため、コンポーネントは一切表示されません。ユーザーは、アクセス制御が設定されたレポートを実行する際に、それらの制御が Oracle9iAS Portal で設定され、そこから提供されることを意識しません。

Reports Server 構成ファイルにおける *security* 構成エントリは、次のようになります。

```
<security id="rwSec" class="oracle.reports.server.RWSecurity">
  <property name="securityuserid" value="portal_id/portal_password@portal_schema"
    confidential="yes" encrypted="no"/>
</security>
```

注意： 独自のセキュリティ・インタフェースを実装する場合は、Java で実装する必要があります。Oracle9iAS Reports Services API の詳細は、Oracle Technology Network (<http://otn.oracle.com/> 英語サイト) または Oracle Technology Network Japan (<http://otn.oracle.co.jp/> 日本語サイト) を参照してください。

security 要素の有効な属性は、表 5-1 で説明します。

表 5-1 *security* 要素の属性

属性	有効値	事前条件またはデフォルト	説明
id	文字列	必須	特定の <i>security</i> 要素を識別する、構成 XML ファイル内で一意のキーワード。テキスト文字列または数字のどちらでも使用できます (id="rwSec" など)。
class	「説明」を参照してください。	必須	Reports Server Security Java インタフェース (oracle.reports.server.Security) を実装する完全修飾された Java クラス。デフォルト値 (oracle.reports.server.RWSecurity) は、Oracle9iAS に含まれる Oracle9iAS Portal 経由で提供されるセキュリティ機能に依存します。

また、この例の security 要素では、`securityUserId/portal_id/portal_password@portal_schema` という名前 / 値のペアが使用されています。`securityUserId` はプロパティの名前、`portal_id/portal_password@portal_schema` は Portal へのアクセスに必要な管理者レベルの有効なユーザー ID、パスワードおよび SID を表しています。この例には、その他に *confidential* 属性と *encrypted* 属性が含まれています。`confidential="yes"` は、この要素内で値の暗号化が必要であることを示し、`encrypted="no"` は値がまだ暗号化されていないことを示しています。次に Reports Server を起動したときは、値が自動的に暗号化され、`encrypted` が `yes` にリセットされます。

security 構成要素の詳細は、第 3 章の 3.2.1.5 項「*security*」を参照してください。

5.3 Oracle9iAS Portal での Reports ユーザーおよび 名前付きグループの作成

Oracle9iAS Portal のセキュリティ機能を使用してレポートへのアクセスを制御する場合は、すべての Reports ユーザーを Oracle Internet Directory (OID) に登録し、それらすべてのユーザーに Oracle9iAS Portal を通じてセキュリティ権限を割り当てる必要があります。

注意： LDAP 互換ディレクトリにすでに多数のユーザーを入力済みである場合は、Oracle Internet Directory (OID) の機能を使用してディレクトリ間の同期をとることで、ユーザーを 1 人ずつ入力する手間が省けます。OID の Directory Integration Server の詳細は、『Oracle Internet Directory 管理者ガイド』を参照してください。

Portal では、個別のユーザーおよびユーザーの名前付きグループにセキュリティ権限を付与できます。名前付きグループは、アクセス権限を付与する処理を簡素化するために役立ちます。名前付きグループにアクセス権限のセットを割り当てておけば、グループに個別のユーザーを追加するだけで、そのユーザーに権限のセット全体を付与することができます。

次の各項では、Oracle9iAS Portal でユーザーとグループを作成する方法について概説します。関連する項は、次のとおりです。

- デフォルトの Reports 関連グループ
- ユーザーとグループの作成

5.3.1 デフォルトの Reports 関連グループ

Oracle9iAS Portal をインストールすると、Reports 関連グループが自動的に作成されます。これには次のグループが含まれます。

- [RW_BASIC_USER](#)
- [RW_POWER_USER](#)
- [RW_DEVELOPER](#)
- [RW_ADMINISTRATOR](#)

レポートにアクセスするユーザーは、必ずこれらのグループのいずれかに属していなければなりません。各グループには、アクセス権限のセットが割り当てられています。このセットは、必要に応じてカスタマイズできます。これらのグループのメンバーでないユーザーがレポートを実行しようとする、デフォルトでそれらのユーザーに基本ユーザーの権限が割り当てられます。次の各項では、グループとその権限について説明します。

5.3.1.1 RW_BASIC_USER

基本ユーザーには、EXECUTE 権限が割り当てられています。基本ユーザーは、レポートを実行し、結果を参照することができます。セキュリティ・チェックが失敗した場合は、他の Reports ユーザー・グループに比べて簡単なエラー・メッセージが表示されます。たとえば次のようなメッセージです。

```
Security Check Error
```

5.3.1.2 RW_POWER_USER

RW_BASIC_USER グループの権限よりも多くの権限が付与された RW_POWER_USER グループは、基本ユーザーよりも詳細なエラー・メッセージを受け取ります。たとえば、HTML に対する実行が許可されていない場合にそれを実行しようすると、次のようなメッセージが表示されます。

```
Cannot run report to HTML
```

これは、同じエラーに関して RW_BASIC_USER が受け取るメッセージより詳しく記述されています。

5.3.1.3 RW_DEVELOPER

RW_DEVELOPER グループは、RW_POWER_USER グループおよび RW_BASIC_USER グループの権限よりも多くの権限が付与され、システム環境を表示する SHOWENV や SHOWMAP などのコマンドを実行できます。このグループには、テストの実行や詳細なエラー・メッセージの取得を必要とする開発者を割り当てます。

5.3.1.4 RW_ADMINISTRATOR

このグループに割り当てられたユーザーには、MANAGE 権限が付与されます。このユーザーは、Oracle9iAS Portal に登録されているレポート定義ファイル、サーバー、プリンタの各オブジェクトを作成、更新および削除することができます。開発ユーザー向けのすべてのリンクに加えて、管理者は Oracle9iAS Portal でアクセス可能な「Component Management」ページの「アクセス」タブにナビゲートできます。ここで管理者は、特定のレポートにアクセスできるユーザーを指定できます。管理者権限を持つユーザーは、他のユーザーにセキュリティ権限を割り当てたり、Oracle9iAS Reports Services から完全なエラー・メッセージを受け取ることができます。

これらのユーザーは、Oracle9iAS Reports Queue Manager の管理者の機能に対するアクセス権限も持っています。つまり、サーバー内のジョブの再スケジュール、削除、順序変更、およびサーバーの停止を含めた、サーバー・キューの管理を実行できます。

ユーザーの作成と管理の詳細は、『Oracle9i Application Server セキュリティ・ガイド』を参照してください。

5.3.2 ユーザーとグループの作成

Oracle9iAS Portal は、Oracle Internet Directory (OID) に対する Delegated Administration Service (DAS) インタフェースを使用して、Portal にアクセスできるようにユーザーを登録します。Portal を通じて、DAS インタフェースから新しいユーザーを作成することができます。ユーザーとグループの新規作成の詳細は、Oracle9iAS のドキュメントが収録されている CD 内の『Oracle9i Application Server セキュリティ・ガイド』を参照してください。このドキュメントの「Oracle9iAS Portal によるセキュリティの設定」という章を参照してください。

5.4 アクセス制御の設定

ポートレットにアクセスし、アクセス制御の設定に必要なタスクを遂行するためには、事前に Oracle9iAS Portal で十分なレベルの権限を割り当てられていることが必要です。権限が付与されたグループへの参加の詳細は、Oracle9iAS のドキュメントが収録されている CD 内の『Oracle9i Application Server セキュリティ・ガイド』を参照してください。このドキュメントの「Configuring Oracle9iAS Portal Security」という章を参照してください。

十分なレベルの権限を取得したら、この後の項で次の作業について確認してください。

- [使用可能日カレンダーの作成](#)
- [プリンタの登録](#)
- [Reports Server の登録](#)
- [レポートの登録](#)

5.4.1 使用可能日カレンダーの作成

使用可能日カレンダーの定義はオプションのステップで、レポート、サーバーおよびプリンタにアクセスできる時間とできない時間を指定することにより、これらへのアクセスをさらに制限できます。レポート、Reports Server およびプリンタがいつでも利用可能であれば、使用可能日カレンダーは不要です。

この項では、次の項目について説明します。

- シンプルな使用可能日カレンダーの作成
- 結合された使用可能日カレンダーの作成

レポート、Reports Server またはプリンタに関連付けられる使用可能日カレンダーは1つのみです。本番環境で複数の使用可能日ルールが必要な場合は、使用可能日カレンダーを結合できます。

5.4.1.1 シンプルな使用可能日カレンダーの作成

シンプルな使用可能日カレンダーは、1つの使用可能日ルールを定義したものです（たとえば、日曜日～土曜日の午前 12:00 ～午後 10 : 00 など）。

シンプルな使用可能日カレンダーを作成する手順は、次のとおりです。

1. Oracle9iAS Portal に管理者としてログインします。
2. Portal メイン・ページの上部にある「ビルダー」ボタンをクリックします。
3. 表示されたページで、「管理」タブをクリックします。
4. 「Oracle Reports Security」ポートレットの下で、「レポート・カレンダー・アクセスの作成」をクリックします。
5. カレンダーの内部名、表示名およびポータル DB プロバイダを指定します。
 - 「名前」フィールドには、使用可能日カレンダーを Portal 内部で一意に識別する名前 (MY_CALENDAR など) を入力します。この名前は、有効なコンポーネント名に関する Portal のルールに従っていなければなりません。このルールは、Portal のオンライン・ヘルプで説明されています。
 - 「表示名」フィールドには、この使用可能日カレンダーが Portal を通じて公開される際に表示される、このカレンダーの名前を入力します。内部名とは異なり、表示名にはスペースを含めることができます。
 - プロバイダの値リストから「ポータル DB プロバイダ」を選択します。Portal で追加または作成されるコンポーネントは、すべてポータル DB プロバイダに属していなければなりません。このリストには、コンポーネントを作成する権限がある場合に使用できるプロバイダの名前だけが表示されます。

注意： ポータル DB プロバイダの作成の詳細は、Oracle9iAS Portal のオンライン・ヘルプを参照してください。

6. 「次へ」 ボタンをクリックします。
7. オプションとして、「説明」にカレンダーの説明を入力できます。
8. 「次へ」 ボタンをクリックします。
9. 「使用可能日時」 ページで、カレンダーのパラメータを定義します。

「期間」には、1 単位の継続期間を構成する時間の長さを指定します。たとえば、レポートのアクセスを、任意の日の午前 9:00 から午後 5:00 まで許可するカレンダーを設定するとします。この場合、月、日および年については「開始」と「終了」の設定が同じになりますが、時間と分については、「開始」の設定が午前 9:00、「終了」の設定が午後 5:00 になります。この例では、任意の日におけるレポートの使用可能期間が、午前 9:00 ～午後 5:00 になります。

「繰返し」には、継続期間が繰り返される頻度を指定します。

- 「一度のみ繰返し」は、継続期間が繰り返されず、期間が終了すると関連するコンポーネントが利用できなくなることを意味します。たとえば、「一度のみ繰返し」を選択し、1 年の継続期間を設定した場合は、1 年後に関連コンポーネントが利用できなくなります。
- 「毎年」は、継続期間が毎年繰り返されることを意味します。「毎年」を選択し、「期間」の設定で、開始と終了の日付を同じにして、「開始」時間を午前 9:00、「終了」時間を午後 5:00 に設定した場合は、この使用可能日カレンダーに関連付けられている Reports コンポーネントが毎年同じ日の午前 9:00 ～午後 5:00 に利用可能となります。
- 「毎月」は、「期間」に指定された「開始」と「終了」の日付で表される継続期間が毎月繰り返されることを意味します。「毎月」を選択し、「開始」と「終了」の両方に同じ年と日付（2001 年 7 月 25 日）を指定し、「開始」時間を午前 9:00、「終了」時間を午後 5:00 に設定した場合は、関連コンポーネントが毎月 25 日の午前 9:00 ～午後 5:00 に利用可能となります。
- 「日付 / 曜日毎」の設定は、「毎月」のみに適用されます。「日付 / 曜日毎」では、継続期間が特定の日付で設定されるか（たとえば、月の 25 ～ 29 日は常に利用可能とするなど）、または特定の曜日で設定されるか（たとえば、今月 25 ～ 29 日の月曜～金曜は常に利用可能とするなど）を指定します。
- 「毎週」は、「期間」に指定された曜日の期間が毎週繰り返されることを意味します。
- 「毎日」は、「期間」に指定された時間が、毎日繰り返されることを意味します。

- 「頻度」には、繰返し間隔 n （「繰返し」で選択した値に応じて、年数、月数、週数、日数を表す）の値を指定します。たとえば、毎週繰り返すように継続期間を設定し、「頻度」を2に設定すると、継続期間が2週ごとに（つまり隔週）繰り返されます。
- オプションで、「繰返し期限」をチェックして、カレンダーの終了日時を割り当てます。これによって、「繰返し期限」の日時に、すべての関連コンポーネントが利用できなくなります。

カレンダーでは妥当性チェックが実行されないことに注意してください。継続期間が繰返しの設定を超過しても、エラー・メッセージは生成されません。たとえば、10日間の継続期間を設定し、繰返しを毎週に設定した場合は、期間が重複しますが、ユーザーにはその重複が通知されません。

10. 「次へ」 ボタンをクリックします。
11. 「サマリー」 ページで「カレンダーの表示」 ボタンをクリックして、設定した使用可能日カレンダーをプレビューします。設定を変更する場合は、「前へ」 ボタンをクリックして変更を加えます。
12. 「サマリー」 ページで「完了」 ボタンをクリックして、使用可能日カレンダーの作成を完了します。
13. 「閉じる」 ボタンをクリックして、「Oracle Reports Security」 ページに戻ります。

5.4.1.2 結合された使用可能日カレンダーの作成

結合された使用可能日カレンダーは、複数の使用可能日カレンダーを結合して1つの使用可能日カレンダーにしたものです。これは、使用可能期間を設定した後に、その期間から特定の日（休日など）を除外する場合に便利です。

カレンダーを結合する際に、一方のカレンダーのすべての日から、もう一方のカレンダーのすべての日を除外するように指定できます。たとえば、あるカレンダーでは月曜～金曜が使用可能日であり、別のカレンダーでは水曜日のみが使用可能日であるとしめます。この2つのカレンダーを結合し、その際に水曜日のカレンダーを除外することで、結合されたカレンダーでは月曜、火曜、木曜、金曜が使用可能日となります。

あるいは、年全体で平日を使用可能日とするシンプルなカレンダーを作成し、その後で新年の休暇を除外するカレンダー、その次の休暇を除外するカレンダー、さらにその次の休暇を除外するカレンダーなどのシンプルなカレンダーを追加します。これらすべてのカレンダーを結合し、休暇のカレンダーすべてを除外することで、コンポーネントは年間を通じて会社営業日のある一定時間のみ利用可能となります。

使用可能日カレンダーを結合する手順は、次のとおりです。

1. Oracle9iAS Portal に管理者としてログインします。
2. Portal メイン・ページの上部にある「ビルダー」 ボタンをクリックします。
3. 表示されたページで、「管理」 タブをクリックします。

4. 「Oracle Reports Security」ポートレットの下で、「結合されたカレンダーの作成」をクリックします。
5. カレンダーの内部名、表示名およびポータル DB プロバイダを指定します。
 - 「名前」フィールドには、結合された使用可能日カレンダーを Oracle9iAS Portal 内部で一意に識別する名前 (MY_COMBINED_CALENDAR など) を入力します。この名前は、有効なコンポーネント名に関する Portal のルールに従っている必要があります。このルールは、Portal のオンライン・ヘルプで説明されています。
 - 「表示名」フィールドには、この結合された使用可能日カレンダーが Portal を通じて公開される際に表示される、このカレンダーの名前を入力します。内部名とは異なり、表示名にはスペースを含めることができます。
 - プロバイダの値リストから「ポータル DB プロバイダ」を選択します。Portal で追加または作成するコンポーネントは、すべてポータル DB プロバイダに属している必要があります。このリストには、コンポーネントを作成する権限がある場合に使用できるプロバイダの名前のみが表示されます。

注意： ポータル DB プロバイダの作成の詳細は、Oracle9iAS Portal のオンライン・ヘルプを参照してください。

6. 「次へ」ボタンをクリックします。
 7. オプションとして、「説明」にカレンダーの説明を入力できます。
 8. 「次へ」ボタンをクリックします。
 9. 「選択」ページで、「使用可能日カレンダー」リストの中から結合するカレンダーを選択します。
- Windows では、[Ctrl] キーを押しながらクリックして複数のカレンダーを選択します。
- UNIX では、選択する各カレンダーをクリックします。
- このページに表示されるのは、結合された使用可能日カレンダーの作成で使用するポータル DB プロバイダに対してすでに定義されている使用可能日カレンダーです。
10. 右矢印をクリックして、選択したカレンダーを「選択された使用可能日カレンダー」リストに移動します。
 11. 「次へ」ボタンをクリックします。
 12. 「除く」ページで、「使用可能日カレンダー」リストの中から除外する日付を含んだカレンダーを選択します。

Windows では、[Ctrl] キーを押しながらクリックして複数のカレンダーを選択します。

UNIX では、選択する各カレンダーをクリックします。

これらのカレンダーには、使用可能日の指定を取り消す日付が含まれています。

13. 右矢印をクリックして、選択したカレンダーを「除外された使用可能日カレンダー」リストに移動します。
14. 「次へ」ボタンをクリックします。
15. 「サマリー」ページで、「カレンダーの表示」ボタンをクリックして、設定したカレンダーをプレビューします。

除外した箇所が表示されていない場合は、別のビューを選択します。たとえば、月のビューのかわりに週のビューを選択します。

結合を変更する場合は、カレンダーを閉じ、目的のページに戻るまで「前へ」ボタンをクリックします。
16. 「完了」をクリックして、結合されたカレンダーの作成を完了します。

このカレンダーは、他のカレンダーと結合することも、登録されている Reports コンポーネントにそのまま適用することもできます。

5.4.2 プリンタの登録

Oracle9iAS Portal のセキュリティ・フレームワークの中では、プリンタを登録する必要があります。レポートは、Reports Server で利用できるすべてのプリンタで実行することができます。ただし場合によっては、Oracle9iAS Portal ユーザーをそれらのプリンタのサブセットに制限したり、一定期間プリンタの使用を制限したり、特定のレポートの出力を印刷する特別なプリンタを識別することが必要になります。

Portal へのプリンタの登録は、Portal を通じて実行するレポートだけでなく、スタンドアロンの URL から実行するレポートにも意味があります。

Oracle9iAS Portal 内に登録したプリンタは、Oracle9iAS Reports Server に関連付けることができます。多数のプリンタを登録できます。ただし、Portal にレポートを登録し、特定の Oracle9iAS Reports Server を選択する場合は、それらの Reports Server に関連付けられているプリンタのみが印刷に使用できます。

登録されたレポートを送信できるプリンタの登録済サブセットを、さらに制限することができます。たとえば、Oracle9iAS Reports Server が最高経営責任者のオフィスのプリンタに接続されている場合は、最高経営責任者以外の従業員が総勘定元帳レポートを実行する際に、そのプリンタを選択できないようにする必要があります。レポート要求を実行する Oracle9iAS Portal ユーザーに対してプリンタのサブセットを提示し、出力の送信先を選択させることができます。

プリンタを登録する手順は、次のとおりです。

1. Oracle9iAS Portal に管理者としてログインします。
2. Portal メイン・ページの上部にある「ビルダー」ボタンをクリックします。
3. 表示されたページで、「管理」タブをクリックします。

4. 「Oracle Reports Security」ポートレットの下で、「Oracle Reports Security の設定」をクリックします。
5. 「Reports プリンタ・アクセス」ポートレットの下で、「Reports プリンタ・アクセスの作成」をクリックします。
6. 表示されたページでは、「名前」、「表示名」、「ポータル DB プロバイダ」の各フィールドにデフォルト値が入力されています。これらのデフォルト値を必要な値に変更します。
 - 「名前」フィールドには、プリンタを Oracle9iAS Portal 内部で一意に識別する名前 (MY_PRINTER など) を入力します。この名前は、有効なコンポーネント名に関する Portal のルールに従っている必要があります。このルールは、Portal のオンライン・ヘルプで説明されています。
 - 「表示名」フィールドには、プリンタが Portal を通じてユーザーに公開される際に表示される、そのプリンタの名前を入力します。内部名とは異なり、表示名にはスペースを含めることができます。
 - 「ポータル DB プロバイダ」値リストから、プリンタの所有者となるポータル DB プロバイダを選択します。Portal で追加または作成するコンポーネントは、すべてポータル DB プロバイダに属していなければなりません。このリストには、コンポーネントを作成する権限がある場合に使用できるプロバイダの名前だけが表示されます。
7. 「次へ」ボタンをクリックします。
8. 表示されたページで、必要な値を入力します。
 - 「OS プリンタ名」フィールドに、オペレーティング・システムのプリンタ名を入力します。次に例を示します。

UNIX: <printer_name>

Windows: ¥¥<printer_server>¥<printer_name> (for a remote printer)
<printer_name> (for a local printer)

このプリンタは、Reports Server で利用することが必要です。

注意： プリンタの可用性は、Reports Server のホスト・マシンからオペレーティング・システムを通じて設定します。

- オプションで、「説明」フィールドにプリンタの説明を入力します。
9. 「次へ」ボタンをクリックします。
 10. オプションで、使用可能日カレンダーを選択してプリンタが使用可能な日時を制限することができます。使用可能日カレンダーの詳細は、「[使用可能日カレンダーの作成](#)」を参照してください。

11. 「完了」 ボタンをクリックします。

このプリンタに関する Portal の設定を要約したページが表示されます。このページでは、設定を編集することができます。また、プリンタに関する詳細な登録情報を確認したり、設定を Portal から完全に削除することができます。

12. 「閉じる」 ボタンをクリックして、このページを閉じ、「Oracle Reports Security」 ページに戻ります。

これで、Portal へのプリンタの登録が完了しました。この登録は、Portal を通じて実行されるレポートのほかに、Portal の外部から実行されるレポートにも意味があります。

5.4.3 Reports Server の登録

Reports Server に対するアクセス制御を定義するには、事前に Portal 内でサーバーを登録する必要があります（必須）。登録によって、利用可能なすべての Reports Server を特定して検索するために必要な情報が、Portal に提供されます。これは、個別のレポートを登録する際に特に重要となります。レポートの登録では、Reports Server のリストから選択する必要がありますが、登録されているサーバーのみがこのリストに表示されるためです。

この項では、Reports Server を Oracle9iAS Portal に登録する方法について説明します。

Reports Server を登録する手順は、次のとおりです。

1. Oracle9iAS Portal に管理者としてログインします。
2. Portal メイン・ページの上部にある「ビルダー」 ボタンをクリックします。
3. 表示されたページで、「管理」 タブをクリックします。
4. 「Oracle Reports Security」 ポートレットの下で、「Oracle Reports Security の設定」 をクリックします。
5. 「Reports サーバー・アクセスの作成」 をクリックします。
6. 表示されたページでは、「名前」、「表示名」、「ポータル DB プロバイダ」の各フィールドにデフォルト値が入力されています。これらのデフォルト値を必要な値に変更します。
 - 「名前」フィールドには、Reports Server を Oracle9iAS Portal 内部で一意に識別する名前（MY_REPORTS_SERVER など）を入力します。この名前は、有効なコンポーネント名に関する Portal のルールに従っている必要があります。このルールは、Portal のオンライン・ヘルプで説明されています。
 - 「表示名」フィールドには、サーバーが Portal を通じてユーザーに公開される際に表示される、そのサーバーの名前を入力します。内部名とは異なり、表示名にはスペースを含めることができます。

- 「ポータル DB プロバイダ」値リストから、サーバーの所有者となるポータル DB プロバイダを選択します。Portal で追加または作成するコンポーネントは、すべてポータル DB プロバイダに属している必要があります。このリストには、コンポーネントを作成する権限がある場合に使用できるプロバイダの名前だけが表示されます。
- 7. 「次へ」 ボタンをクリックします。
- 8. 「Server Definition」 ページで、次の作業を行います。

- 「Reports サーバー名」 フィールドに、Reports Server の名前を入力します。これはサーバーをインストールする際に各自で付けた一意の名前です。ただし、インストール後に名前を変更した場合は、新しい名前を入力してください。
- オプションとして、「説明」 フィールドに Reports Server の説明を入力します。
- 「Oracle Reports Web Gateway URL for JSP reports」 フィールドに、Java Server Page (JSP) ファイルの位置を示す URL を入力します。次に例を示します。

`http://<your_web_server>.<domain>:<port>/<virtual_path_to_JSPs>/`

注意： 仮想パスの指定の詳細は、[第 3 章「Oracle9iAS Reports Services の構成」](#)を参照してください。

- 「Oracle Reports Web Gateway URL for RDF reports」 フィールドに、Reports Servlet の位置を示す URL を入力します。次に例を示します。

`http://<your_web_server>.<domain>:<port>/<virtual_path_to_rwservlet>/rwservlet`

注意： 仮想パスの指定の詳細は、[第 3 章「Oracle9iAS Reports Services の構成」](#)を参照してください。

- 要求があったときに、Portal に登録されているレポート定義ファイルのみを実行する場合は、「Run Only Registered Report Definition Files」チェック・ボックスをチェックします。

このボックスをチェックしないと、この Reports Server は、Portal に登録されていないものも含めたあらゆるレポート定義ファイルを受け付けます。ただし、レポート要求を送信するユーザーが、この Reports Server に対するアクセス権限を持っている場合に限りです。

- 「プリンタ」リストから、この Reports Server で使用できるプリンタを選択します。複数のプリンタを選択できます。

Windows では、[Ctrl] キーを押しながら選択するプリンタをクリックします。

UNIX では、選択する各プリンタをクリックします。

9. 「次へ」 ボタンをクリックします。
10. 「宛先タイプ」 ページで、この Reports Server のカスタム宛先タイプを入力します。
Oracle9iAS に設定されているデフォルトの宛先を使用する場合は、Portal での追加の設定が必要ないので注意してください。

注意： カスタム宛先タイプの詳細は、第 4 章「[Oracle9iAS Reports Services の宛先の構成](#)」を参照してください。

11. 「次へ」 ボタンをクリックします。
12. オプションとして、「使用可能日カレンダー」 ページで、使用可能日カレンダーを選択し、この Reports Server でレポート要求を受け付けることのできる日時を制御します。
13. 「完了」 ボタンをクリックします。
14. 「閉じる」 ボタンをクリックします。

これで、「Oracle Reports Security の設定」 ページに戻ります。

これで、Oracle9iAS Reports Server の登録が完了しました。次は、レポートを登録します。

5.4.4 レポートの登録

レポートの登録は必須のステップです。この登録によって、レポートを実行できるユーザー、レポートの実行時間、レポート要求の処理に使用するサーバー、レポートの配布方法、およびレポートの送信先プリンタを定義できます。

登録では、レポートにアクセスできるユーザーを指定するほかに、Portal パラメータ・フォームを通じて、ユーザーとレポートの対話方法を指定することもできます。

ユーザーは、Reports Builder でユーザー・パラメータを作成します。次にユーザーは、Portal でこれらのパラメータの名前を指定し、エンド・ユーザーがレポートを実行する際に、それらのパラメータの値を選択または入力できるようにします。設計時に Reports Builder で定義されなかったパラメータは、Portal で設定しても、実行時に Reports Server によって無視されます。

Portal を通じて利用できるパラメータ設定を使用して、設計時に Reports Builder で定義したパラメータのサブセットを複製または作成します。こうして、Oracle9iAS Portal を通じてレポートを実行する際に、パラメータの範囲を取得します。

Oracle9iAS Portal 内でレポートを登録すると、Oracle9iAS Portal コンポーネントが作成されます。これは、Portal を通じてポートレットとして展開できます。Portal では、レポート・ファイルのインスタンスを 1 つだけ登録することをお勧めします。1 つのレポートに複数の Portal Reports オブジェクトを定義すると、実行時にそのすべてのオブジェクトがセキュリティ・チェックを受けます。このうち 1 つでもセキュリティ・チェックに合格しなかった場合は、すべてのオブジェクトが不合格となり、ジョブは実行されません。

レポートを登録する手順は、次のとおりです。

1. Oracle9iAS Portal に管理者としてログインします。
2. Portal メイン・ページの上にある「ビルダー」ボタンをクリックします。
3. 表示されたページで、「管理」タブをクリックします。
4. 「Oracle Reports Security」ポートレットの下で、「Oracle Reports Security の設定」をクリックします。
5. 「レポート定義ファイル・アクセス」ポートレットの下で、「レポート定義ファイル・アクセスの追加」をクリックします。
6. 「名前」、「表示名」、「ポータル DB プロバイダ」の各フィールドには、デフォルト値が入力されています。これらのデフォルト値を必要な値に変更します。
 - 「名前」フィールドには、レポートを Portal 内部で一意に識別する名前（MY_REPORT など）を入力します。この名前は、有効なコンポーネント名に関する Portal のルールに従っていなければなりません。このルールは、Portal のオンライン・ヘルプで説明されています。
 - 「表示名」フィールドには、レポートが Portal を通じて公開される際に表示される、このレポートの名前を入力します。内部名とは異なり、表示名にはスペースを含めることができます。
 - 「ポータル DB プロバイダ」値リストから、レポートの所有者となるポータル DB プロバイダを選択します。Portal で追加または作成されるコンポーネントは、すべてポータル DB プロバイダに属していなければなりません。このリストには、コンポーネントを作成する権限がある場合に使用できるプロバイダの名前だけが表示されます。
7. 「次へ」ボタンをクリックします。
8. 次のように情報を入力または選択します。
 - 「Reports サーバー」値リストから、このレポートの実行に使用できる Reports Server を選択します。

複数のサーバーを選択する方法：

Windows では、[Ctrl] キーを押しながら各サーバーをクリックします。

UNIX では、各サーバーをクリックします。

- 「Oracle Reports ファイル名」フィールドに、Builder でレポートに付けた名前を、その拡張子も含めて入力します。

レポート定義ファイルは、.rdf ファイル、.jsp ファイル、.xml ファイルのいずれかです。このファイルへのパスが REPORTS_PATH 環境変数に含まれている場合は、ここでパスを入力しないでください。パスが REPORTS_PATH 環境変数に含まれていない場合は、ここでファイル名とともにパスを入力します。この作業は、スタンドアロンの JSP として実行するものを除くすべてのレポート定義ファイルに対して実行します。

- オプションとして、このレポートの説明を「説明」フィールドに入力します。
- 「実行」フィールドで、「サーブレットを使用」または「as JSP」のいずれかを選択します。

Reports Servlet を介してレポートを実行する場合は、「サーブレットを使用」を選択します。Reports Servlet を介さずに、スタンドアロンで JSP レポートを実行する場合は、「as JSP」を選択します。

ここでの選択は、次のウィザード・ページの選択肢に影響します。

9. 「次へ」ボタンをクリックします。
10. 「必須パラメータ」ページで、必要なランタイム情報を設定します。これらの設定は、Reports Servlet を介して実行する場合のみ適用されます。[Ctrl] キーを押しながらクリックして複数選択を指定した箇所では、実行時に値リストがユーザーに表示され、その中からユーザーが各自のランタイム情報を選択できるようになります。
 - 「タイプ」では、このレポートで受け付ける宛先タイプを指定します。キャッシュ、ファイル、メール、Oracle Portal、プリンタ、またはカスタム宛先タイプの中から選択します。Portal で Reports Server を登録するときに指定したカスタム宛先タイプを、このレポートに関連付けるサーバーがサポートする場合は、指定したタイプがこのリストに表示されます。
 - 「書式」では、このレポートで受け付ける出力フォーマットを定義します。HTML、HTMLCSS、PDF、XML、RTF、デリミタ付き、PostScript、および文字の中から選択します。
 - 「プリンタ」では、このレポートを送信できる登録済プリンタを指定します。このリストに表示されるプリンタは、このレポートに関連付ける Reports Server へのアクセスを設定する際に選択したプリンタによって決まります。登録されているプリンタすべてが表示されます。ユーザーがランタイム・パラメータ・フォームで Reports Server を選択する場合は、選択した Reports Server に関連付けられているプリンタおよびそのユーザーがアクセスできるプリンタのみが表示されます。
 - 「パラメータ・フォーム・テンプレート」では、レポートを実行する際に使用する Portal パラメータ・フォームのルック・アンド・フィールを定義するテンプレートを指定します。この値は、レポートが Portal を通じて公開される場合のみ使用されます。テンプレートは値リストから選択します。選択したテンプレートの外観を確認するには、「テンプレートのプレビュー」をクリックします。

注意： このリストに独自のテンプレートを追加する方法の詳細は、**Oracle9iAS Portal** のオンライン・ヘルプを参照してください。

11. 「次へ」 ボタンをクリックします。
12. 「オプションのパラメータ」 ページで、レポートの既存パラメータに対する制限を定義します。
 - 「名前」 は、ユーザーが指定できる値を制限するシステム・パラメータまたはユーザー・パラメータの名前（SALES_REGION や COPIES など）です。
 - 「表示名」 は、ランタイム・パラメータ・フォームでパラメータを識別するために使用する名前です。
 - 「LOV」 は、パラメータ・フォームに含められる事前定義の値リストの名前です。このリストは事前に作成しなければなりません。値リストの作成の詳細は、**Portal** のオンライン・ヘルプを参照してください。
 - 「下限」 は、値の範囲に対して設定する下限値です。
 - 「上限」 は、値の範囲に対して設定する上限値です。
 - パラメータと値を追加するために行を増やす場合は、「パラメータ追加」 をクリックします。
13. 「次へ」 ボタンをクリックします。
14. オプションで、使用可能日カレンダーの名前を入力します（またはカレンダーのリストから選択）。

使用可能日カレンダーを使用して、このレポートを実行できる日時を制限します。詳細は、[5.4.1 項「使用可能日カレンダーの作成」](#)を参照してください。
15. 「次へ」 ボタンをクリックします。
16. オプションで、有効化トリガーを入力してプログラムによる制限を作成します。

有効化トリガーは、条件付き制限を作成するために使用されます。条件付き制限とは、「必須パラメータ」 ページにも「オプションのパラメータ」 ページにも定義できない制限です。有効化トリガーは PL/SQL ファンクションです。

有効化トリガーとして指定するファンクションは、ブール値（TRUE または FALSE）を返す必要があります。ファンクションが TRUE を返した場合は、ジョブが実行されます。ファンクションが FALSE を返した場合は、エラー・メッセージが表示され、ジョブは実行されません。
17. 「完了」 をクリックして、ウィザードを閉じ、レポートの登録を完了します。

結果のページには、登録情報の要約が表示されます。このページで、レポートに追加操作を実行するかどうかを選択できます。

- レポートのランタイム・パラメータ・フォームを表示するには、「カスタマイズ」をクリックします。

表 5-2 では、このページで指定可能なオプションをまとめています。

表 5-2 ランタイム・パラメータ・フォームのオプション

オプション	説明
レポートの実行	指定したパラメータ値でこのレポートを実行する場合にクリックします。
パラメータの保存	パラメータ値の選択内容を保存する場合にクリックします。
サーバー	このレポート要求を受け取る Oracle Reports Server を選択します。「レポート名およびサーバー」ページで選択したサーバーのみが、このリスト・ボックスに表示されます。
プリンタ	レポート出力を印刷するプリンタを選択します。「必須パラメータ」ページで選択したプリンタのみが、このリスト・ボックスに表示されます。
Destype	宛先タイプを選択します。「必須パラメータ」ページで選択した宛先タイプのみが、このリスト・ボックスに表示されます。
Desformat	宛先フォーマットを選択します。「必須パラメータ」ページで選択した宛先フォーマットのみが、このリスト・ボックスに表示されます。
Desname	Destype が FILE の場合は出力ファイルの名前を入力し、Destype が MAIL の場合は電子メール・アドレスを入力します。複数のアドレスはカンマで区切ります。宛先名は、Destype に FILE または MAIL を選択した場合に必要となります。
SSOCONN	1 つ以上の SSO 接続文字列を入力します。複数の文字列は、スペースではなくカンマで区切ります。
ユーザーに表示	ユーザーがこのレポート要求を実行するときにランタイム・パラメータ・フォームに表示する各パラメータをチェックします。ボックスをチェックしないパラメータは、ユーザーに表示されません。
CGI/Servlet コマンド・キー	オプションで、このレポートに対して実行するコマンドラインを識別する cgicmd.dat ファイルのキーを入力します。
追加されたユーザー・パラメータ	このフィールドは、追加のユーザー・パラメータを入力するために使用します。たとえば、このフィールドを使用して、このレポートの配布方法を定義する配布 XML ファイルのパスと名前を入力します。 その場合、コマンドライン要求または cgicmd.dat ファイル内でこれらの値を指定する際の構文を使用します。複数の追加パラメータを入力する場合は、各エントリをスペースで区切るだけです。 配布 XML ファイルの詳細は、 第 9 章「拡張配布の作成」 を参照してください。

Reports Server のクラスタ

クラスタとは、複数のサーバーを 1 つのコミュニティに仮想的にグループ化したものです。このコミュニティの目的は、要求処理をクラスタの各メンバーに効率的に分散することです。Oracle9iAS Reports Services では、ピアレベルのクラスタ化を採用しています。つまり、受信要求の分散と処理に対して、クラスタの全メンバーが同等の役割を担います。このため、あるメンバーが停止しても、残りのメンバーが要求負荷の管理を続行します。またあるメンバーのキャッシュに存在する出力を、他のメンバーが使用することもできます。1 台のマシンの障害が原因でシステム全体がダウンする単一の障害箇所はありません。

この章では、クラスタへのサーバーの登録に関する情報、およびサーバーのクラスタ化による利点について説明します。この章には、次の項があります。

- [クラスタの概要](#)
- [クラスタのセットアップ](#)

6.1 クラスタの概要

たとえば、次のようなサーバーがあるとします。

```
serverA.cluster1  
serverB  
serverC.cluster1
```

ServerA.cluster1 と serverC.cluster1 は、cluster1 と呼ばれる同じクラスタのメンバーです。これらのサーバーは、クライアントからの要求を協同して処理します。クライアントが同期要求を serverA.cluster1 に送信したときに、このサーバーにそのジョブ・タイプのアイドル・エンジンがない場合、このサーバーは、serverC.cluster1 で処理できるかどうかをチェックします。serverC.cluster1 にアイドル・エンジンがあれば、serverA.cluster1 は、処理のために要求を serverC.cluster1 に渡します。

この例では、スタンドアロン・サーバーである ServerB は、処理要求を他のサーバーから受け取ることも、他のサーバーに送信することもできません。

1 つのクラスタにいくつでもサーバーを追加できます。あるクラスタ・メンバーが停止すると、未処理の同期ジョブはクラスタ内の他のサーバーにリダイレクトされます。クラスタ内で 1 つでもサーバーが稼動していれば、クラスタは処理を続けます。

クラスタが次のスケジュール済要求または即時要求を処理するメンバーを決定する際、次の基準に従って優先順位を付けます。

1. クラスタ内のいずれかのサーバーに、要求と一致する情報がキャッシュされているか。
2. 現在、同様のジョブがキューにあるか。
3. そのジョブ・タイプのアイドル・エンジンが利用可能か。
4. 現在アクティブなエンジンの数は、そのジョブ・タイプに対してサーバーで指定されている MAXENGINE の数に達していないか。

スタンドアロンのサーバーとクラスタ化されているサーバーの基本構成は、どちらも同じです。クラスタでは、共通のクラスタ名と共通の公開鍵および秘密鍵を共有すればよいだけで、特別な構成要件はありません。使用するプラットフォーム、クラスタ内のサーバー数、あるいはサーバーの位置には何の制約也没有ありません。クラスタ・サーバー間でリソースを共有する必要もありません。

エンジン出力はクラスタ内の特定の Reports Server でローカルにキャッシュされますが、クラスタ全体でその存在を認識しており、それを使用できます。あるサーバーがダウンすると、そのサーバーのキャッシュ・ファイルは再利用できなくなります。つまり、出力を取得するには、クラスタ内の別のサーバーが要求に応答しなければなりません。キャッシュは永続的に格納されるので、サーバーが再開すると、すべてのキャッシュ・ファイルが利用可能になります。

6.2 クラスタのセットアップ

Oracle9iAS Reports Services でのクラスタ化は簡単に行うことができます。つまり、<server_name>.cluster または <server_name>.xyz のように、すべてのメンバー・サーバーに同じドット拡張子を持つ名前を付け、すべてのクラスタ・メンバーが同じ公開鍵および秘密鍵を共有すればよいだけです。

この項では、Reports Server 名の変更、公開鍵および秘密鍵の作成と指定、およびクラスタへの要求送信について説明します。ここには、次の項があります。

- Reports Server 名の変更
- 公開鍵および秘密鍵の新規生成
- 公開鍵および秘密鍵のサーバー構成ファイルへの入力
- Reports Server の再起動
- クラスタへの要求送信

6.2.1 Reports Server 名の変更

この項では、少なくとも 1 つの Reports Server がすでにセットアップされているものとします。この場合、クラスタ名をサーバー名に追加するには、サーバー名を変更する必要があります。

注意： サーバーがインストールされていない場合は、インストール時に、同じクラスタ名を持つ別々の名前を各サーバーに付けます。たとえば、servernameA.cluster1、servernameB.cluster1 などです。

Reports Server 名を変更する手順は、次のとおりです。

1. サーバーが実行中の場合は、停止します。
 - Windows NT のサービスとして実行中の場合は、「サービス」コントロール・パネルで停止します。
 - Windows NT のサーバー実行可能ファイルまたは UNIX のシェル・スクリプト経由で実行中の場合は、「Oracle Reports Server」ダイアログ・ボックスの「停止」ボタンをクリックします。
 - Windows NT または UNIX のコマンドラインから実行中の場合は、コマンド・プロンプトで次の Windows または UNIX のコマンドを入力します。

次のコマンドによって、サーバーが正常に停止します。

```
rwserver server=<server> shutdown=normal authid=<admin/pword>
```

次のコマンドによって、サーバーが直ちに停止します。

```
rwserver server=<server> shutdown=immediate authid=<admin/pword>
```

次のコマンドによって、関連メッセージを表示せずにサーバーが停止します。

```
rwserver server=<server> shutdown=normal authid=<admin/pword> batch=yes
```

rwserver コマンドで使用されるキーワードは、付録 A「コマンドライン引数」で説明します。

2. Reports Server 構成ファイル (<server_name>.conf) にカスタムの構成設定が含まれている場合、このファイル名を新しいクラスタ・サーバー名 (<server_name>.<cluster_name>.conf) に変更します。

構成ファイルは、UNIX および Windows 上の次のパスにあります。

```
ORACLE_HOME¥reports¥conf¥<server_name>.conf
```

Reports Server 構成ファイルにカスタムの構成設定が含まれていない場合は、名前を変更したサーバーを再起動すると、新しい名前を持つ新しい構成ファイルが自動的に生成されます。

3. 影響を受けるすべてのファイルで Reports Server 名を変更し、各クラスタ・メンバーに同じクラスタ名を指定します。

- サブレット構成ファイル (rwservlet.properties) をオープンし、クラスタ名を含むようサーバー名を変更します。次に例を示します。

```
SERVER=<server_name>.<cluster_name>
```

サブレット構成ファイルは、Windows NT および UNIX 上の次のパスにあります。

```
ORACLE_HOME¥reports¥conf¥rwservlet.properties
```

- Windows NT サービスとして実行しているサーバー名を変更する場合は、サービスをアンインストールしてから再びインストールする必要があります。

Windows NT サービスをアンインストールするには、コマンド・プロンプトで次のように入力します。

```
rwserver -uninstall <server_name>
```

Windows NT サービスを再びインストールするには、コマンド・プロンプトで次のように入力します。

```
rwserver -install <server_name>.<cluster_name>
```

注意： 再インストール後、サーバーは自動的に起動します。すべてのサーバー・クラスタ・メンバー名の変更が終わるまでサーバーを停止させておき、クラスタのセットアップが完了した時点ですべて一緒に起動することもできます。サーバーの停止の詳細は、[第2章「Oracle9iAS Reports Services の起動および停止」](#)を参照してください。

Reports Server を再起動する前に、サーバーの公開鍵および秘密鍵を生成し、その情報を各メンバー・サーバーの構成ファイルに入力できます。この手順は、次の各項で説明します。

6.2.2 公開鍵および秘密鍵の新規生成

サーバーの公開鍵および秘密鍵のファイルは、クラスタ・メンバー間のメッセージ暗号化および認証に使用されます。デフォルトのファイルは、次のパスにある `rwrn.jar` ファイルに格納されています（UNIX と Windows の両方で共通）。

```
ORACLE_HOME¥reports¥jlib¥rwrn.jar
```

クラスタの各メンバーの構成ファイル（`<server_name>.<cluster_name>.conf`）には、同じ公開鍵および秘密鍵のファイルを指定する必要があります。クラスタ・メンバーが排他的な公開鍵および秘密鍵のファイルを共有できるようにするには、クラスタをセットアップする際に新しいバージョンのファイルを生成します。クラスタ・メンバー以外のサーバーは、Oracle9iAS Reports Services で提供されるデフォルトの鍵を使用して処理を続行できます。

新しい公開鍵および秘密鍵のファイルを生成するには、コマンド・プロンプトで次のコマンドを入力します。

```
java oracle.reports.utility.KeyManager <public_key_file_name> <private_key_file_name>
```

これらのファイルを特定のディレクトリに生成するには、コマンドラインで新しい公開鍵および秘密鍵のファイルの名前とともに必要なパスを指定します。コマンドラインでファイル名のみを指定すると、鍵ファイルは現行ディレクトリに生成されます。

6.2.3 公開鍵および秘密鍵のサーバー構成ファイルへの入力

新しい公開鍵および秘密鍵のファイルを生成した後、その情報をクラスタの全メンバーの Reports Server 構成ファイルに入力する必要があります。このファイルの各クラスタ・メンバーのバージョンは、UNIX と Windows の両方とも、各サーバーのホスト・マシン上の次のパスに置かれています。

```
ORACLE_HOME¥reports¥conf¥<server_name>.<cluster_name>.conf
```

公開鍵および秘密鍵のファイルを変更するには、サーバー構成ファイルの *connection* 要素で、次のように *cluster* サブ要素のエントリを変更または追加します。

```
<cluster publicKeyFile="path and filename of new public key" privateKeyFile="path  
and filename of new private key">
```

connection 要素の詳細は、[第 3 章「Oracle9iAS Reports Services の構成」](#)を参照してください。

6.2.4 Reports Server の再起動

クラスタ・メンバー名を変更し、共通の公開鍵および秘密鍵をそれぞれに指定した後、Reports Server を再起動してクラスタを有効化できます。

Reports Server を再起動する手順は、次のとおりです。

- Reports Server を Windows NT サービスとして起動する場合は、「サービス」コントロール・パネルでサービスを起動します。
- Reports Server をコマンドラインから起動する場合は、コマンド・プロンプトで次のように入力します。

Windows NT の場合：

```
rwserver server=<server_name>.<cluster_name>
```

UNIX の場合：

```
rwserver.sh server=<server_name>.<cluster_name>
```

クラスタ・メンバー名を変更し、公開鍵および秘密鍵を再指定し、Reports Server を再起動したら、クラスタのセットアップ・プロセスは完了です。

6.2.5 クラスタへの要求送信

クラスタに要求を送信する手順は、次のとおりです。

Reports Servlet または JSP で、次のように指定します。

```
server=<cluster_name>
```

たとえば、mercury.cluster1、venus.cluster1 という名前の 2 台のクラスタ・マシンがある場合、server を次のように指定します。

```
server=cluster1
```

Reports Servlet または JSP は、クラスタ内で実行中の Reports Server を検出し、その Reports Server に要求を送信します。キャッシュとの一致状況、あるいはサーバーの負荷状況によって、その Reports Server は要求を処理するか、クラスタ内の別のサーバーにリダイレクトします。

データ・ソース・シングル・サインオン

Oracle9iAS Reports Services の利点の 1 つに、トランSPORTAブル・データ・ソースがあります。これによって、ユーザーが 1 つのジョブを実行する際に複数のデータ・ソースにログインする必要がなくなります。これは、シングル・サインオン (SSO) を通じて実現できます。

SSO を使用すると、各ユーザーに一意の識別子を設定し、それをそのユーザー固有のリソースと結び付けることができます。ユーザーのリソースには、キーで識別される接続文字列が含まれており、これを使用して様々なデータ・ソースにアクセスします。ユーザーは、セッションごとに 1 回のログインで一意に識別されます。このログインによって、ユーザーのリソースを参照してそのユーザーが適切なデータ・ソースにアクセスできるようにするため、ユーザーはこの情報を自ら入力する必要がありません。

SSO は、Oracle9iAS Reports Services、Oracle Internet Directory および Oracle Login Server との連携によって可能となります。これらはすべて、Oracle9i Application Server に付属しています。

SSO は、セキュアなサーバー環境でのみ実装されます。つまり、Reports で SSO の実装を検討する場合、Reports Server 構成ファイルにセキュリティ・ポリシーが正しく設定されていることが前提となります。

注意： セキュリティ設定については、次の各章で取り上げています。第 3 章「Oracle9iAS Reports Services の構成」では、サーバーのセキュリティ・ポリシーを定義する Java クラスを指定する方法を説明し、第 5 章「ユーザー・アクセスの制御」では、Oracle9iAS Portal を通じてセキュリティ設定をサーバー、プリンタおよびレポートに適用する方法を説明しています。また、付録 A「コマンドライン引数」では、SSOCONN コマンドライン引数に関する情報を提供しています。

SSO を使用する場合は、管理者が各ユーザーにユーザー ID を設定します。この作業は、Oracle Internet Directory (OID) で行われ、その際に OID のユーザー・インタフェースである Delegated Administration Service (DAS) または Oracle9iAS Portal (Portal にユーザーを登録すると、その情報が OID に保存されます) が使用されます。

ユーザー ID は、ユーザー名とパスワードで構成されます。ユーザーの設定が終わると、管理者はそれらのユーザーにリソースを割り当てることができます。リソースは、様々なデータ・ソースへの接続文字列から構成されます。ユーザーは、ログイン時にユーザー名とパスワード（ユーザー ID）を入力し、それによってそのユーザー ID に関連付けられたすべてのリソースにアクセスできるようになります。Oracle Login Server からセッション Cookie が発行されます。これは、そのセッションに許可されているすべての入口を開くためのキーとして効果的に機能します。

この章では、データ・ソース SSO について説明します。その中で、SSO 環境のアーキテクチャに関する情報を提供するとともに、SSO のユーザー・リソース（接続文字列）を設定する際に役立つヒントを示します。

注意： SSO 関連コンポーネント（Oracle Internet Directory など）を設定するための要件と手順の詳細は、『Oracle Internet Directory 管理者ガイド』および『Oracle9i Application Server Oracle HTTP Server 管理ガイド』を参照してください。

この章の主な項は、次のとおりです。

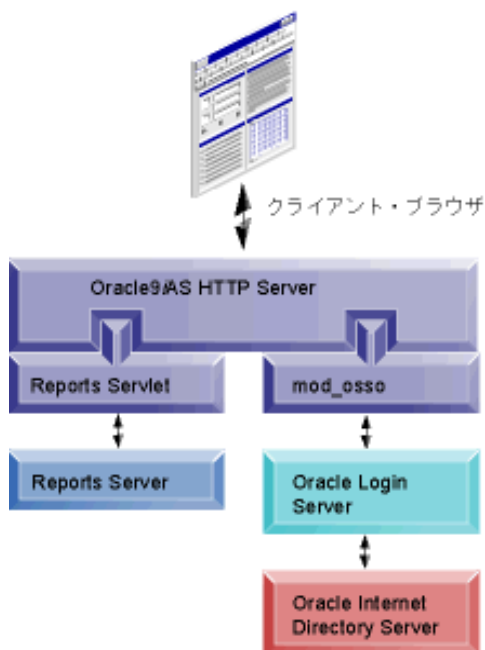
- [SSO のアーキテクチャ](#)
- [ユーザーの接続文字列を設定する方法](#)

7.1 SSO のアーキテクチャ

7.1.1 SSO のコンポーネント

図 7-1 は、SSO のコンポーネント・アーキテクチャの概要です。

図 7-1 SSO のアーキテクチャ



SSO 環境のコンポーネントは、次のとおりです。

- クライアント Web ブラウザ
- Oracle HTTP Server

SSO のコンテキストでは、Oracle HTTP Server がクライアントと mod_osso 間の仲介役として機能します。

- mod_osso モジュール

mod_osso モジュールは、Oracle Login Server のパートナー・アプリケーションで、認証サポートを提供します。Reports 環境では、このモジュールが Reports Servlet と Oracle Login Server 間の仲介役として機能します。

- Reports Servlet

Reports Servlet は、Oracle9iAS Reports Services のコンポーネントで、Oracle HTTP Server の Oracle9iAS Containers for Java 2 Enterprise Edition (OC4J) の内部で実行されます。SSO のコンテキストでは、Reports Servlet が mod_osso と Reports Server 間の仲介役として機能します。

- Reports Server

Reports Server は、クライアント要求を処理します。この処理には、認証と許可の確認、スケジューリング、キャッシュおよび配布が含まれます。SSO のコンテキストでは、Reports Server から Reports Servlet に対して、その Reports Server がセキュアかどうか通知されます。

- Oracle Single Sign-on (SSO Server)

Oracle SSO Server は、ユーザーのシングル・サインオン・セッションを管理します。mod_osso モジュールは、Reports Servlet と Oracle SSO Server 間の仲介役となります。mod_osso モジュールによって、認証要求がサーバーに渡されます。サーバーは、Oracle Internet Directory (OID) でユーザーを検索することで、ユーザーのログイン資格証明を検証します。

- Oracle Internet Directory (OID)

OID は、ユーザー情報のリポジトリです。ユーザー情報には、自宅の住所、電子メール、電話番号などの個人情報のほかに、ユーザー ID や接続文字列などのアクセス情報が含まれます。OID は、Oracle 固有の Lightweight Directory Access Protocol version 3 (LDAPv3) サービスです。このサービスは、Oracle9i リレーショナル・データベースの上にアプリケーションとして構築されています。Oracle で製造するすべての LDAP 対応製品 (Oracle9i、Oracle9iAS および Oracle9iAS Portal を含む) は、サポートされている Oracle Internet Directory クライアントとして出荷されます。

- Oracle Delegated Administration Service (DAS)

DAS は、OID 内のユーザーとグループを管理するためのユーザー・インタフェース・サービスです。DAS インタフェースを使用して、ユーザーと接続文字列の情報を OID に入力します。DAS は、OID クライアントとして作成されたサブリットです。DAS には、インストール時に設定した URL を通じてブラウザからアクセスします。

7.1.2 SSO のトランザクション

SSO 環境のトランザクションは、次のように処理されます。

1. クライアント・ブラウザを通じて Oracle HTTP Server に要求が送信されます。
2. HTTP Server が、この要求を (ジョブのランタイム URL とともに) Reports Servlet に渡します。Reports Servlet は、Reports Server を呼び出して、Reports Server がセキュアであることを確認します。

3. Reports Server がサーブレットに応答します。応答が yes である場合は、Reports Servlet から mod_osso に認証を要求するメッセージ 401 が送信され、応答が no の場合は、Reports Servlet が要求を処理します。
4. mod_osso モジュールが、メッセージ 401 を受け取って Oracle Login Server に接続し、ユーザーが認証済みであるかどうかを確認します。
5. Login Server が応答します。応答が no である場合は、Login Server がクライアントにログイン画面を送信し、yes の場合は、元の URL 要求がユーザーの ID とともに承認され、その要求が Reports Servlet によって処理されます。
6. クライアントがログイン情報を Login Server に送信します。Login Server は、そのログイン情報を Oracle Internet Directory 内の情報と照合します。
7. ログイン情報が正しい場合は、元の URL 要求が承認され、処理が完了します。ログイン情報が正しくない場合は、エラーが返され、クライアントに再試行のプロンプトが表示されるか、または処理が停止します。再試行回数の上限は、SSO Server の構成で指定されます。

7.2 ユーザーの接続文字列を設定する方法

Oracle Internet Directory (OID) には、LDAP ソースから OID にユーザーをバッチ・ロードするためのツールはありますが、それらのユーザーの接続文字列（ユーザーがデータ・ソースへアクセスするために必要なパスワードとスキーマ ID）をバッチ・ロードするツールは今のところありません。このため、この情報を手動で入力するか、または入力を自動処理するプロシージャを作成する必要があります（経験のある LDAP プログラマであれば、OID にリソースを取り込むプロシージャを作成できます）。

ユーザー・ベースが大規模である場合は、この作業が困難となる可能性があります。

ただし、各ユーザーが自分の接続文字列情報を入力する方法がいくつかあります。この項では、これらの方法について概説します。

7.2.1 初期要件

どちらの方法も、ユーザーが OID にすでに入力されていることが必要です。OID を初めて使用する場合でも、他の LDAP データ・ソースにすでにユーザー・ベースが入力されていれば、OID で提供するツールを使用してそれらのユーザーをバッチ・ロードできます。

注意： バッチ・ロードの詳細は、『Oracle Internet Directory 管理者ガイド』を参照してください。

ユーザーを入力した LDAP データ・ソースがない場合は、手動でユーザーを入力する必要があります。

7.2.2 方法 1: ユーザーに OID へのアクセスを許可する

ユーザーに自分の接続文字列情報を入力させる 1 つ目の方法は、ユーザーに OID へのアクセスを許可する方法です。OID へのユーザー・インタフェースは、Oracle Delegated Administration Service (DAS) と呼ばれます。

注意： ユーザーが DAS にアクセスするには、あらかじめ管理者が各ユーザーのユーザー ID を OID に入力しておく必要があります。この作業は、他のソースの LDAP ディレクトリにすでに入力されている情報をバッチ・ロードすることで実行できます。

バッチ・ロードの詳細は、『Oracle Internet Directory 管理者ガイド』を参照してください。

Oracle9iAS をインストールする際に、DAS の場所を指定します。ユーザーに DAS へのアクセスを許可する場合は、この場所を示す URL を各ユーザーに通知します。

DAS にアクセスしたユーザーは、「Extended Preferences」タブの「Resource Assignment」セクションで各自の情報を入力できます。この場合、リソースが入力されていないと、「Extended Preferences」タブがユーザーに表示されないなどの問題が発生することがあります。

ユーザー・ベースに対してデフォルトのリソースを入力することもできますが、これも時間がかかる可能性があります。

7.2.3 方法 2: 接続文字列を割り当ててログイン時にユーザーに入力させる

2 つ目の方法は、安全性と効率性で最初の方法より優れていると考えられます。安全性が高いのは、ユーザーが DAS に直接入力する必要がないためです。効率性が高いのは、レポート要求を送信する処理の一環として情報が入力されるためです。

この方法では、ユーザーがデータ・ソースへ最初にログインするときに、各自の情報を入力します。

OID 管理者は、各ユーザーにレポートへの URL を電子メールで通知します。各電子メールには、受信者用の一意のパスワードとスキーマ接続文字列のほかに、その接続文字列を使用するための指示が記載されます。

注意： 複数のデータ・ソースを利用するレポートの場合は、複数の接続文字列を提供する必要があります。この項で説明した方法を使用して、複数の接続文字列を 1 通の電子メールで送信するか、または接続文字列ごとに 1 通ずつ電子メールを送信できます。

URL には、SSOCONN コマンドライン・オプションが含まれています。これは、まだ存在しない接続キーを呼び出すものです。次に例を示します。

```
ssoconn=mykey1/DB,mykey2/PDSApp
```

各 URL は、同じ接続キー（mykey1 など）を呼び出すことができます。これは、各ユーザーの入力するデータが、キー名ではなく一意の情報であるためです。

ユーザーはプロンプトに応じて接続情報を入力し、その情報が自動的に OID に保存されます。

将来的には、ユーザーがレポートを実行する際に、ユーザー ID（ユーザー名とパスワード）の入力を要求されるようになる予定です。これによってユーザー ID を含んだ Cookie が生成され、それが Reports Servlet に送信されて、OID 内の接続文字列情報（リソース）を取得するようになります。

第 II 部

サーバーへの要求送信

第 II 部では、レポート公開に関する有用な詳細情報を説明します。これには、要求の実行方法、高度な自動レポート配布の設定方法、XML カスタマイズ・ファイルを使用して実行時にレポートをカスタマイズする方法、データベース・トリガーを使用してレポートを自動的に起動する方法などが含まれます。

第 II 部には、次の章があります。

- [第 8 章「レポート要求の実行」](#)
- [第 9 章「拡張配布の作成」](#)
- [第 10 章「XML によるレポートのカスタマイズ」](#)
- [第 11 章「イベント・ドリブンによる公開」](#)

レポート要求の実行

この章では、Reports Server にレポート要求を送信する様々な方法について説明します。この章には、次の項があります。

- [Reports の URL 構文](#)
- [レポート要求の方法](#)
- [Oracle9iAS Portal でのレポート・ポートレットの公開](#)
- [Web ブラウザからのレポート要求の指定](#)
- [URL エンジンへの要求送信](#)
- [自動実行のためのレポートのスケジューリング](#)
- [キャッシュからのレポート出力の再使用](#)
- [キー・マップ・ファイルの使用方法](#)

8.1 Reports の URL 構文

この項では、レポートを公開するための URL の作成に関するクイック・リファレンス情報を提供します。ここで取り上げる配布タイプは、次の 3 つです。

- サブレット
- JSP
- CGI（下位互換性を確保するため）

ここで提供する情報は、Windows 環境と UNIX 環境でほとんど同じです。相違点も示してあります。

8.1.1 サブレット

Reports Servlet を経由したレポート実行の URL 構文は、次のとおりです。

```
http://<web_server>.<domain_name>:<port>/<alias>/rwservlet?<parameters>
```

表 8-1 では、サブレットの URL の構成要素を説明します。

表 8-1 Reports Servlet を呼び出す URL の構成要素

構成要素	説明
<web_server>	Oracle HTTP Server をインストールしたときに各自で付けた名前。
<domain_name>	所属する組織のドメイン名。
<port>	Oracle HTTP Server が要求をリスニングするポート番号。ポートが指定されない場合は、デフォルト（80）が使用されます。
<alias>	URL がアクセスするファイルへの絶対パスにかわる仮想パス。
rwservlet	Oracle9iAS Reports Services サブレットを起動します。
?	コマンドライン引数の開始を示します。
<parameters>	すべてのコマンドライン引数、またはコマンドライン引数が指定されているキー・マップ・ファイルへのキー。

Reports Servlet を呼び出す URL は、次のようになります。

```
http://neptune.world.com:80/reports/rwservlet?keyname
```

keyname は、cgicmd.dat ファイル内の一意のヘッダー（キー名）の下に示されているコマンドラインを参照します。JSP ファイルの場合は、この処理が異なるので注意してください。JSP ファイルでは、キーワード / 値のペア（cmdkey=value）を使用して、cgicmd.dat ファイル内のコマンドラインに対するキー名を指定します。キー・マッピングの使用の詳細は、8.9 項「キー・マップ・ファイルの使用法」を参照してください。

JSP ファイルに Web レイアウトとペーパー・レイアウトの両方が含まれている場合に、サーブレットの使用によって Reports の JSP ファイルが使用できなくなることはありません。レポートを実行する際に、URL にサーブレットを指定し、report=<myreport>.jsp というコマンドライン引数で JSP を呼び出します。

次に例を示します。

```
http://neptune.world.com:80/reports/rwservlet?report=myreport.jsp&destype=cache&desformat=html
```

コマンドラインの詳細は、付録 A「コマンドライン引数」を参照してください。

注意： これらのパラメータは、JSP ファイルの内部に指定することもできます。

8.1.2 JSP

JSP ベース・レポートの URL の構文は、次のとおりです。

```
http://<web_server>.<domain_name>:<port>/<alias>/myreport.jsp?<parameters>
```

表 8-2 では、JSP ベース・レポートの URL の構成要素を説明します。

表 8-2 JSP ベース・レポートの URL の構成要素

構成要素	説明
<web_server>	Oracle HTTP Server をインストールしたときに各自で付けた名前。
<domain_name>	所属する組織のドメイン名。
<port>	Oracle HTTP Server が要求をリスニングするポート番号。ポートが指定されない場合は、デフォルト（80）が使用されます。
<alias>	URL がアクセスするファイルへの絶対パスにかわる仮想パス。
myreport.jsp	この URL で実行されるレポートの *.jsp ファイル。
?	コマンドライン引数の開始を示します。
<parameters>	すべてのコマンドライン引数、またはコマンドライン引数が指定されているキー・マップ・ファイルへのキー（あるいはその両方）。

JSP ベース・レポートの起動に使用される URL は、次のようになります。

```
http://neptune.world.com:80/jsp/myreport.jsp?
```

URL には、追加のコマンドライン・パラメータを含んだ `cgicmd.dat` ファイル内のコマンドラインを参照するキーを指定できます。その場合は、名前と値のペア (`cmdkey=keyname`) を使用する必要があります。これは、疑問符で示される問合せ文字列の開始点の後であれば、URL の任意の位置に指定できます。次に例を示します。

```
http://neptune.world.com:80/jsp/myreport.jsp?userid=scott/tiger@hrdb&cmdkey=key1
```

URL でパラメータを結合するときは、スペースを一切挿入せずに、アンパサンド (&) を使用します。

JSP を使用する場合に、Reports Servlet を同時に使用できないわけではありません。レポートを実行する際に、URL にサーブレットを指定し、次のコマンドライン引数で JSP を呼び出します。report=<myreport>.jsp

次に例を示します。

```
http://neptune.world.com:80/reports/rwervlet?report=myreport.jsp&destype=cache&desformat=html
```

コマンドライン・キーワードの詳細は、[付録 A「コマンドライン引数」](#)を参照してください。cgicmd.dat ファイルの詳細は、[8.9 項「キー・マップ・ファイルの使用方法」](#)を参照してください。JSP レポートを実行する際に Reports Servlet を使用するかどうかについては、[第 1 章「Oracle9iAS Reports Services のアーキテクチャ」](#)を参照してください。

8.1.3 CGI

注意： Reports CGI は、下位互換性を確保する目的で Oracle9iAS Reports Services に用意されている機能です。レポートを配布するときは、サーブレット実装または JSP 実装のいずれかを使用する方法を強くお勧めします。

Windows で Reports CGI を経由してレポートを実行する URL の構文は、次のとおりです。

```
http://<web_server>.<domain_name>:<port>/<alias>/rwcgi.exe?<parameters>
```

UNIX では、次のようになります。

```
http://<web_server>.<domain_name>:<port>/<alias>/rwcgi.sh?<parameters>
```

[表 8-3](#) では、CGI ベース・レポートの URL の構成要素を説明します。

表 8-3 Reports CGI を呼び出す URL の構成要素

構成要素	説明
<web_server>	Oracle HTTP Server をインストールしたときに各自で付けた名前。
<domain_name>	所属する組織のドメイン名。
<port>	Oracle HTTP Server が要求をリスニングするポート番号。ポートが指定されない場合は、デフォルト（80）が使用されます。
<alias>	URL がアクセスするファイルへの絶対パスにかわる仮想パス。
rwcgi.exe	Oracle9iAS Reports Services の CGI コンポーネントを起動する実行可能ファイル。UNIX マシンに Reports Services がインストールされている場合は、".exe" のかわりに ".sh" を使用します。
?	コマンドライン引数の開始を示します。
<parameters>	すべてのコマンドライン引数、またはコマンドライン引数が指定されているキー・マップ・ファイルへのキー。

Windows で CGI 実装の起動に使用される URL は、次のようになります。

http://neptune.world.com:80/CGI-BIN/rwcgi.exe?key2

UNIX では、次のようになります。

http://neptune.world.com:80/CGI-BIN/rwcgi.sh?key2

UNIX マシンに Reports Services がインストールされている場合は、".exe" のかわりに ".sh" を使用します。次に例を示します。

http://neptune.world.com:80/CGI-BIN/rwcgi/sh?key2

8.2 レポート要求の方法

レポート要求は、様々な方法で実行することができます。次に各方法について説明します。

■ rwclient コマンドライン

rwclient コマンドライン（UNIX では rwclient.sh）は、非 Web アーキテクチャ内でコマンドラインからレポート要求を実行するために使用します。このコマンドラインは、コマンドラインを解析して指定の Oracle9iAS Reports Server に送る実行可能ファイルを参照します。このコマンドラインでは、Oracle9iAS Reports Runtime の実行可能ファイル rwrun（UNIX では rwrun.sh）で 사용되는ものと似たコマンドライン引数を使用できます。

Windows での典型的な `rwclient` コマンドライン要求は、次のようになります。

```
rwclient report=<my_report>.rdf userid=<username>/<password>@<my_db>  
server=<server_name> destype=html desformat=cache
```

UNIX では、同じコマンドを次のように記述します。

```
rwclient.sh report=<my_report>.rdf userid=<username>/<password>@<my_db>  
server=<server_name> destype=html desformat=cache
```

コマンドライン引数の詳細は、[付録 A「コマンドライン引数」](#)を参照してください。

■ URL

ブラウザからレポートを実行する場合は、URL 構文を使用します。この URL 構文は、Oracle9iAS Reports Servlet（および下位互換性のための CGI）によって `rwclient` コマンドライン要求に変換され、その後 Oracle9iAS Reports Services によって処理されます。ユーザーのブラウザからレポート要求を行うのに必要な URL 構文をユーザーに提供するか、または URL 構文をハイパーリンクとして Web サイトに追加することができます。この方法については、この後の項で詳しく説明します。

■ Oracle9iAS Portal 経由

Oracle9iAS Portal コンポーネントを使用すると、Oracle9iAS Portal ページまたはポートレット内にレポートへのリンクを追加するか、またはレポートの結果を直接ポートレットに出力することができます。各レポート・リンクは、レポート要求に関する情報を含んだパッケージ・プロシージャを示します。Oracle9iAS Reports Services のシステム管理者は、Oracle9iAS Portal のウィザードを使用してパッケージ・プロシージャを作成することで、Web を通じてレポートをより便利で安全に公開できるようにします。

Oracle9iAS Portal ページ・グループにアクセスする権限のあるユーザーは、リンクをクリックするだけでレポートを実行できます。システム管理者はウィザードから直接レポートを実行できます。詳細は、Oracle9iAS Portal のオンライン・ヘルプを参照してください。

レポートをポートレットとして公開する方法の詳細は、「[Oracle9iAS Portal でのレポート・ポートレットの公開](#)」を参照してください。

■ パッケージ・プロシージャ

SRW.RUN_REPORT は、Oracle9iAS Reports Runtime のコマンドを実行するビルトインです。SRW.RUN_REPORT を指定するときは、SERVER 引数を Oracle9iAS Reports Services サーバーの名前に設定して、SRW.RUN_REPORT コマンドが `rwclient` コマンドと同じように動作するようにします。

詳細は、Oracle9i Developer Suite (Oracle9iDS) Reports Builder のオンライン・ヘルプを参照してください。

8.3 Oracle9iAS Portal でのレポート・ポートレットの公開

レポートを公開する最適な方法の 1 つが、Oracle9iAS Portal のセキュアな宣言型インタフェースを使用する方法です。ポータル内でレポートを公開するには、次の作業が必要です。

1. **レポートのプロバイダを作成します。**このステップでは、ポータル内でユーザーに提供するレポートが収納されるプロバイダを定義します。
2. **レポート定義ファイル・アクセスを作成します。**このステップでは、レポートのプロパティ、特にレポートを収納するプロバイダを定義することで、ページ設計者がレポートをポートレットとして利用できるようにします。
3. **レポート・ポートレットをページに追加し、必要に応じてカスタマイズします。**このステップでは、ユーザーがレポートをページ上のポートレットとして利用できるようにするとともに、ページ設計者がレポート・パラメータを設定し、レポートの自動実行をスケジューリングできるようにします。

8.3.1 レポートに対するプロバイダの作成

レポートを収納するプロバイダをまだ定義していない場合は、これを作成する必要があります。プロバイダの作成の詳細は、Oracle9iAS Portal のオンライン・ヘルプを参照してください。

注意： レポートを収納するプロバイダは、データベース・プロバイダである必要があります。また、その「アクセス」ページで「プロバイダとして公開」の設定が選択されていることが必要です。

8.3.2 レポート定義ファイル・アクセスの作成

レポートをポートレットとして提供するには、次の作業が必要です。

注意： 多数のレポートに対してレポート定義ファイル・アクセスを作成する必要がある場合は、それらをバッチ登録する方法が効率的です。詳細は、[付録 C 「Oracle9iAS Portal へのレポートのバッチ登録」](#)を参照してください。

1. Oracle9iAS Portal ホーム・ページで、「企業ドキュメント」タブをクリックします。
2. 「ビルダー」をクリックします。
3. 「管理」タブをクリックします。

4. 「Oracle Reports Security」ポートレットで、「Oracle Reports Security の設定」をクリックします。
5. 「レポート定義ファイル・アクセス」ポートレットで、「レポート定義ファイル・アクセスの追加」をクリックします。
6. ウィザードのステップに従います。設定に関する情報が必要な場合は、右上端にある疑問符をクリックしてください。ウィザードの最後に、「完了」をクリックします。
7. 「アクセス」タブをクリックします。
8. 「ポータルに公開」をクリックします。
9. 「適用」をクリックします。これでレポートがポートレット・リポジトリに追加されたので、レポートをページに追加することができます。

8.3.3 ページへのレポート・ポートレットの追加

ポートレット・リポジトリに追加されたレポートのポートレットは、他のポートレットと同様に任意のページに追加できます。

1. Oracle9iAS Portal ホーム・ページで、「企業ドキュメント」タブをクリックします。
2. 「ビルダー」をクリックします。
3. 「構築」タブをクリックします。
4. 「ページ・グループ」ポートレットで、作成したレポート・ポートレットを配置するページ・グループの名前を「ページ・グループ名」に入力します。
5. 「ページの作成」をクリックして新しいページを作成するか、既存のページの名前を入力して「編集」をクリックすることでそのページを編集します。
6. 新しいページを作成する場合は、ウィザードのステップに従います。設定に関する情報が必要な場合は、右上端にある疑問符をクリックしてください。作業が完了したら、「完了」をクリックします。既存のページを編集する場合は、次のステップに進みます。
7. レポート・ポートレットを追加するページ領域で、「ポートレットの追加」ツールをクリックします（**ヒント**: ツールにマウスを合せると、そのツールに関するヒントが表示されます）。
8. ポートレット・リポジトリを、レポート・ポートレットを含んでいるプロバイダまでドリルダウンします。レポート・ポートレットは、ポートレット・リポジトリの中の、そのポートレットが属しているポータル DB プロバイダの下に表示されます。プロバイダの位置は、ポートレット・リポジトリがどのように編成されているかによって異なります。ポータル DB プロバイダはかなり新しいプロバイダであるため、ポートレット・リポジトリの「新規」ページの下に表示される可能性があります。
9. レポート・ポートレットの名前をクリックして、「選択したポートレット」リストに追加します。

10. 「OK」をクリックします。
11. レポート・ポートレットの右上端にある「カスタマイズ」をクリックします。
12. 「パラメータ」タブにパラメータ値を入力し、必要に応じて「スケジュール」タブでジョブの自動実行をスケジュールリングします。

8.4 Web ブラウザからのレポート要求の指定

レポート要求に必要な URL 構文をユーザーに提供できます。あるいは、URL 構文をハイパーリンクとして Web ページに追加することもできます。

注意： レポートを Web サイトで公開するもう 1 つの方法に、Oracle9iAS Portal コンポーネントを作成する方法があります。詳細は、Oracle9iAS Portal のオンライン・ヘルプを参照してください。

URL 構文は、次の形式で提示できます。

- 完全な URL 要求。次に例を示します。

```
http://<your_webserver>.<domain_name>:<port>/<alias>/rwservlet?
report=myreport.rdf&USERID=<username>/<password>@<my_db>
&SERVER=<server_name>&DESFORMAT=html&DESTYPE=cache
```

他のコマンドライン引数が必要な場合は、rwclient の有効なコマンドライン引数のリストが掲載されている [付録 A「コマンドライン引数」](#) を参照してください。

- キー・マッピングを使用して単純化した URL 要求。次に例を示します。

```
http://<your_webserver>.<domain_name>:<port>/<alias>/rwservlet?key1
```

URL 構文をハイパーリンクとして Web ページに追加する手順は、次のとおりです。

1. ホスト Web ページで、次のように HREF タグを設定します。

```
<A HREF="http://<web_server>.<domain_name>:<port>/<alias>/rwservlet?key1">
Employee Directory</A>
```

2. リンクをホスティングする Web ページをユーザーに表示します。
3. ユーザーはこのリンクをクリックしてレポートを実行します。

8.5 URL エンジンへの要求送信

Reports Server の URL エンジンを実態にさせた場合は、次のコマンドライン引数を使用してジョブ要求を URL エンジンに送信できます。

- `urlParameter` は、キャッシュに取り込まれる URL を指定します。たとえば、`http://www.oracle.com` や JSP レポートなどです。
- `jobType` は、URL エンジンに関連付けられているサーバー構成ファイルでのジョブ・タイプの名前 (`urlEngine` など) です。

注意： URL エンジンの実態化の詳細は、第 3 章「[Oracle9iAS Reports Services の構成](#)」を参照してください。

たとえば、`urlParameter` に外部 URL を指定する要求は、次のようになります。

```
http://localhost.com/servlet/RWServlet?server=ReportsServer
+jobType=urlEngine+urlParameter="http://www.oracle.com"
+destype=mail+desname=foo@bar.com+desformat=htmlcss
```

また、`urlParameter` に JSP レポートを指定する要求は、次のようになります。

```
http://<localhost>/servlet/RWServlet?server=ReportsServer+jobType=rwurl
+destype=cache+urlParameter="http%3A%2F%2F<localhost>%2Ffoo.jsp%3Fuserid
%3Dscott%2Ftiger%3Fserver%3DreportsServer"
```

注意： URL に特殊文字が含まれている場合は、`x-www-form-urlencoded` 形式でその URL をエンコードする必要があります。

8.6 自動実行のためのレポートのスケジューリング

サーバーを使用して、Oracle9iAS Reports Queue Manager または Oracle9iAS Portal から、あるいは `SCHEDULE` コマンドライン引数によって、レポートを自動的に実行することができます。スケジューリング機能では、レポートを実行する時刻と頻度を指定できます。

レポートのスケジューリングの詳細は、Oracle9iAS Reports Queue Manager のオンライン・ヘルプを参照してください。

レポートを Oracle9iAS Portal ページで Portal コンポーネントとして公開する場合は、レポート要求が自動的に実行されるようにスケジューリングし、出力されたレポートを指定のページに送信できます。詳細は、Oracle9iAS Portal のオンライン・ヘルプを参照してください。

`SCHEDULE` キーワードは、`rwclient`、`rwservlet`、`rwcgi` の各コマンドで使用できます。詳細は、付録 A の A.4.83 項「[SCHEDULE](#)」を参照してください。

8.7 追加のパラメータ

Reports Server に要求を送信すると、次に示す追加のパラメータが要求とともに暗黙的に送信されます。これらのパラメータの値は変更できません。

表 8-4 レポート要求とともに渡される追加のパラメータ

名前	説明
ACCEPT_LANGUAGE	ブラウザ / ユーザーが受け付ける言語をカンマで区切ったリスト。
REMOTE_ADDR	要求の送信元であるリモート IP アドレス。
REMOTE_HOST	要求の送信元であるリモート・ホスト名。
SCRIPT_NAME	実行されるスクリプトの仮想パス。
SERVER_NAME	Reports Server が実行されているサーバーのホスト名または IP アドレス。
SERVER_PORT	Reports Server が実行されているサーバーのポート番号。
SERVER_PROTOCOL	要求の送信に使用された情報プロトコルの名前とリビジョン。
USER_AGENT	リモート・クライアントのブラウザの説明。

8.8 キャッシュからのレポート出力の再利用

レポートを実行すると、レポート出力のコピーが Oracle9iAS Reports Services のキャッシュに保存されます。その後で、同じキャッシュ・キーを使用する同一のレポートが実行されると、現行要求は重複ジョブとして認識されます。

次に、Reports のキャッシュ機能の効果的な使用例をいくつか紹介します。

- 新規のジョブ要求 "A" が Reports Server に送信された時点で、別のジョブ "B" が現行ジョブ・キュー内で同じキャッシュ・キーを使用している（エンジンが利用可能になるのを待っているか、または実行中）場合は、ジョブ "A" がジョブ "B" からの出力を使用します。

ジョブのキャッシュ・キーには、destype、desname、server、tolerance の各パラメータを除くほとんどすべてのパラメータが含まれています。

このレベルのキャッシュは、自動的に行われます。コマンドラインにその他のパラメータを指定する必要はありません。

- ユーザーが新規のジョブ要求 "A" で tolerance=n (n は分単位) を指定し、そのジョブが発生しなかった場合は、Reports Server が、終了したジョブ・キュー内で n 分以内に正常終了したジョブを検索しようとします。Reports Server がこのようなジョブを検出できた場合は、新規のジョブ要求 "A" がジョブ "B" の出力を返します。

注意： TOLERANCE コマンドライン引数の詳細は、[付録 A「コマンドライン引数」](#)を参照してください。

- クラスタ環境では、クラスタの全メンバーを対象に重複ジョブ（キャッシュ・キーが同じであるジョブ）のチェックが実行されます。同じクラスタ内の別のサーバーで重複ジョブが検出された場合は、ジョブ要求がそのサーバーに転送され、キャッシュされている結果が取得されます。

Oracle9iAS Reports Services のキャッシュの結果は永続的です。Reports Server を停止して再び起動すると、以前のキャッシュの結果がすべて復元され、再び使用できるようになります。

8.8.1 使用上の注意

- キャッシュのサイズは、Oracle9iAS Reports Queue Manager またはサーバー構成ファイル (<server_name>.conf) の cache 要素を通じて設定できます。Oracle9iAS Reports Server は、キャッシュ・ファイルの合計サイズを、設定されたサイズ以下に保つために、使用頻度の最も少ないファイルから順にキャッシュから削除します。さらに、Oracle9iAS Reports Queue Manager を使用してキャッシュを空にすることもできます。

キャッシュの設定の詳細は、Oracle9iAS Reports Queue Manager のオンライン・ヘルプ、および第 3 章「[Oracle9iAS Reports Services の構成](#)」を参照してください。

- レポートの処理中に同一のジョブが送信された場合は、TOLERANCE が設定されていない場合やゼロに設定されている場合でも、Oracle9iAS Reports Services は現在実行されているジョブの出力を再利用します。たとえば、job_1 の実行中に、別のユーザーが job_1 と同じキャッシュ・キーを持つ job_2 を送信したとします。Oracle9iAS Reports Services は、job_1 の出力を job_2 に使用します。この場合、job_2 は独立して実行されずにキャッシュからフェッチされるため、その処理時間はきわめて速くなります。

8.9 キー・マップ・ファイルの使用方法

URL をユーザーに提供する場合、あるいはハイパーリンクを Web サイトに追加する場合は、キー・マップ・ファイルを使用して、URL 要求内のパラメータを単純化したり隠したりできます。

キー・マップ・ファイルには、レポートを実行するコマンド文字列が含まれており、それぞれのコマンド文字列の先頭には一意のキー識別子が付加されています。レポートを JSP として実行する場合を除いて、ランタイム URL ではこのキーのみを参照します。このキー値はサーバーまたはサーブレットによってマップ・ファイル (cgicmd.dat) に送信され、このファイルから指定のキーに関連付けられたコマンドがサーバーまたはサーブレットに返されて処理が進められます。キー・マッピングを使用することによって、すべてのコマンドライン引数をユーザーから隠蔽できます。

キー・マッピングは次の場合に便利です。

- URL を短縮して、使いやすくする場合
- 元の URL を変更することなく、ランタイム・コマンドをマッピングし直す場合
- 企業にとって典型的ないくつかの実行構成を標準化する場合
- ユーザーからパラメータの一部（データベース接続文字列など）を隠蔽する場合
- ユーザーがレポートの実行に使用できるパラメータを制限する場合

キー・マップ・ファイル（`cgicmd.dat`）からキー名を指定するときは、そのキー名をレポート要求 URL の問合せ文字列の開始位置（疑問符の後）に配置する必要があります。ただし、`cmdkey` コマンドライン・キーワードの値としてキー名を指定する場合（`cmdkey=keyname`）は例外です。この場合は、レポート要求 URL における問合せ文字列の任意の位置にキー名を配置できます。`cmdkey` キーワードは、JSP として実行されるジョブおよび `rwServlet` コマンドで使用できます。

注意： `cmdkey` キーワードの詳細は、[付録 A「コマンドライン引数」](#)を参照してください。

8.9.1 キー・マッピングを使用可能にする方法

キー・マッピングは、次のいずれかの条件が満たされたときに使用可能となります。

- 標準のファイル名（`cgicmd.dat`）を持つ有効なファイルが、Web サーバー・マシン（Windows または UNIX）上の `%ORACLE_HOME%\reports\conf\` ディレクトリ（デフォルトの場所）にある。
- Reports Servlet 構成ファイル（`rwServlet.properties`）の `KEYMAPFILE` パラメータの下に、有効なキー・マップ・ファイルが入力されている。
- `rwcgi` が使用される場合に、Web サーバー上の `REPORTS_CGIMAP` 環境変数に有効なキー・マップ・ファイルの名前が指定されている。詳細は、[付録 B「Reports 関連の環境変数」](#)を参照してください。

使用上の注意

- `rwcgi` URL では、最初の引数（疑問符の後の最初の情報）が、名前 / 値のペアの一部でない限り、キーとして扱われます。最初の引数が名前 / 値のペア（つまりキーワード = 値）である場合は、`cgicmd.dat` キー・エントリのかわりにコマンドライン全体が使用されます。

8.9.2 キー・マップ・ファイルへのキー・マッピング・エントリの追加

キー・マップ・ファイルにキー・マッピング・エントリを追加する手順は、次のとおりです。

1. Reports Server をホスティングするマシン上の `cgicmd.dat` ファイルにナビゲートし、これをテキスト・エディタで開きます。

このファイルは、Windows と UNIX のどちらの場合も、次のディレクトリにあります。

```
%ORACLE_HOME%\reports\server\conf\cgicmd.dat
```

2. キー・マッピング・エントリを追加します。次に例を示します。

```
key1: report=<your_report.rdf> USERID=<username>/<password>@<my_db>  
DESFORMAT=html SERVER=<server_name>.<cluster_name (if present)> DESTYPE=cache
```

この例の `key1` は、キーの名前です。

ファイル自体に記述されている特殊なパラメータを除いて、コマンドライン引数は `rwclient` の構文規則に従います。詳細は、[付録 A「コマンドライン引数」](#) を参照してください。

3. Web ページでハイパーリンクを更新します。

詳細は、[8.4 項「Web ブラウザからのレポート要求の指定」](#) を参照してください。

8.9.3 JSP 以外のすべてでキーを使用する場合

レポート要求 URL にキー名を指定するときは、そのキー名が問合せ文字列内の最初の値（疑問符の直後）になるように配置する必要があります。次に例を示します。

```
http://.../rwservlet?keyname
```

次に示すのは、パラメータ・フォームを使用した制限付きの実行に対するキー・マッピングの例です。

URL は次のようになります。

```
http://<web_server>.<domain_name>:<port>/CGI-BIN/rwcgi.exe?key&par1&par2&parN
```

キー・マップ・ファイルには、次のような行が含まれます。

```
KEY: REPORT=<myreport> DEPTNO=%1 MYPARAM=%2 %*
```

これによって、次のコマンドライン要求に相当する要求が生成されます。

```
rwclient REPORT=<myreport> DEPTNO=par1 MYPARAM=par2 parN
```

8.9.4 JSP としてのレポート実行でのキーの使用方法

レポートを JSP として実行する場合に、cgicmd.dat ファイル内のコマンド・キーを呼び出すときは、URL で cmdkey キーワードを使用する必要があります。たとえば、JSP URL は次のようになります。

```
http://.../myreport.jsp?cmdkey=key
```

注意： rwservlet コマンドで *cmdkey* を使用することもできます。

JSP または rwservlet で cmdkey を使用するときは、問合せ文字列内の任意の位置に cmdkey を配置できます。次に例を示します。

```
http://.../example.jsp?parameter1=value1&cmdkey=keyname  
http://.../rwservlet?parameter1=value1&cmdkey=keyname
```

拡張配布の作成

レポートに拡張配布を定義する場合は、配布 XML ファイルの作成を通じて配布を設計できます。このファイルには、レポートのどのセクションがどの出力フォーマットによってどの宛先に送信されるかを指定できます。1つの配布 XML ファイルに、各自で設計したカスタムの（トランスポート可能な）宛先を含めた複数の異なる宛先を指定できます。

この章では、配布 XML ファイルの作成について説明するとともに、いくつかの使用例を示します。この章の主な項は、次のとおりです。

- [配布の概要](#)
- [配布 XML ファイルの概要](#)
- [配布 XML ファイルの要素](#)
- [配布 XML ファイルの例](#)
- [カスタム / Pluggable Destinations に対する XSL 変換](#)

9.1 配布の概要

配布 XML ファイルは、レポート出力の配布を指定するために必須ではありませんが、配布が複雑である場合に役立ちます。たとえば、あるレポートの出力を様々な方法で発行する場合があります。レポートのエグゼクティブ・サマリーを上級管理職に、詳細な内訳を個別の部門管理者にメールで送信するとします。この場合、2つのレポート・セクションから成る1つのレポートを作成します。1つは縦長の概要セクション、もう1つは横長の詳細セクションです。管理者のリストを含んだデータ・モデル・グループに詳細セクションを関連付けただけで、グループの各インスタンスで宛先を変更し、各部門の出力を関連する管理者に送信します。

配布 XML ファイルを使用すると、ある特定のレポートの複数の出力を1つの XML ファイルに定義し、そのファイルをコマンドラインまたは URL から呼び出すことができるため、配布の複雑さが軽減されます。

9.2 配布 XML ファイルの概要

9.2.1 distribution.dtd ファイル

配布 XML ファイルを作成する際は、次のディレクトリ（Windows と UNIX で共通のパス）にある `distribution.dtd` ファイルに定義されている構文に従います。

```
%ORACLE_HOME%\reports\dtd
```

次の各項では様々な要素や属性が説明されているので、あらかじめ `distribution.dtd` ファイルを印刷し、それを参照しながら読み進めると参考になります。

注意： 配布 XML ファイル内では、データの大文字と小文字が区別されます。各種の要素および属性の大文字と小文字は、`distribution.dtd` ファイルで指定されているとおりに記述する必要があります。

`distribution.dtd` ファイルには、配布 XML ファイル内で有効なすべての要素が示されます。これらの要素にはそれぞれ属性があります。デフォルト値がある属性は、その値を変更する必要がある限り、指定しなくてもかまいません。

多数の様々な属性に変数値を使用すると、動的な配布を作成できます。変数値は、配布 XML ファイルを使用するレポート内の列を参照します。

9.2.2 属性内での変数の使用に関する概略

属性内で変数を使用するには、静的値の位置に `&column_name` または `<column_name>` を入力します。

注意： アンパサンド (&) と小なり記号 (<) は、XML で特別の意味を持つ一方で、Oracle9i Reports Developer の特定のコマンドライン引数に必要な記号でもあります (たとえば、文字パラメータにはアンパサンド記号が必要です)。変数を設定する際に、XML におけるこれらの記号の意味と競合しないようにするには、エンコード形式のアンパサンド (`&`) および小なり記号と大なり記号 (`<` と `>`) を指定します。次に例を示します。

XML ファイル内で間違ってコーディングされた変数は次のようになります。

```
<mail id="a1" to="&lt;manager>@mycompany.com" ...
```

XML ファイル内で正しくコーディングされた変数は次のようになります。

```
<mail id="a1" to="&amp;&lt;manager&gt;@mycompany.com" ...>
```

変数での大なり記号 (>) の使用については特に制約がありませんが、一貫性を持たせるために、エンコード形式 (`>`) の使用をお勧めします。

使用する変数の構文は、値が単独で表されるか、または他の値や文字列と組み合わせて表されるかによって異なります。たとえば、mail 要素の "to" 属性の値は、次のいずれかで表されます。

```
<mail id="a2" to="&amp;email" ...>
```

または

```
<mail id="a3" to="&amp;&lt;first_name&gt;.&amp;&lt;last_name&gt;@myco.com ...>
```

最初の例 (id="a2") では、変数が参照する列 (email) に完全なメール・アドレスが含まれているため、追加情報が必要ありません。2 番目の例 (id="a3") では、変数値 (first_name と last_name) と静的テキストを組み合わせで電子メール・アドレスを表しています (静的テキストは first_name の後のピリオドと @myco.com)。どちらの場合も、電子メールのアドレスが動的に指定されます。どちらの例を使用するかは、変数に必要なすべての情報が含まれているか、または補完のための追加情報を必要とするかによって異なります。

さらに複雑なレイアウトの場合は、PL/SQL 式で作成したレポート列を参照することもできます。たとえば、レポートの中で次の PL/SQL 列を定義するとします。

```
PL/SQL formula CF_MAILID: return(:first_name||'.'||:last_name)
```

この列を、配布 XML ファイルで次のように参照します。

```
to="&&lt;CF_MAILID&>;@mycompany.com"
```

9.3 配布 XML ファイルの要素

配布 XML ファイルの要素は、次のとおりです。

- [destinations](#)
- [foreach](#)
- [mail](#)
- [body](#)
- [include](#)
- [file](#)
- [printer](#)
- [destype](#)
- [attach](#)
- [property](#)

このうちほとんどの要素には、その要素の動作を定義する属性があります。次の各項では、配布 XML ファイルの要素と関連する属性について説明します。[9.4 項](#)では、配布 XML ファイルの要素と属性の使用例を示します。

9.3.1 destinations

例

```
<destinations>  
  [One or more distribution specifications]  
</destinations>
```

必須 / オプション

必須です。配布 XML ファイルには、1 つの *destinations* 要素を指定する必要があります。

説明

destinations 要素は、配布 XML ファイルのコンテンツ領域を開始および終了します。配布 XML ファイルのタグ階層で見ると、*destinations* 要素にそれ以外のすべての要素が従属しています。

destinations 要素には、次のサブ要素があります。

- [foreach](#)
- [mail](#)
- [file](#)
- [printer](#)
- [destype](#)

これらの要素について、次の各項で説明します。

9.3.2 foreach

例

```
<foreach>
  <mail id="a1" to="my_addressee@mycompany.com" subject="Fourth Quarter Results">
    <attach format="pdf" name="dept_&department_ID&gt;.pdf" srcType="report"
      instance="this">
      <include src="mainSection"/>
    </attach>
  </mail>
</foreach>
```

OR

```
<mail id="a4" to="recipient@mycompany.com" subject="Regional Results">
  <foreach>
    <attach format="pdf" name="report.pdf" srcType="report" instance="all">
      <include src="mainSection"/>
    </attach>
  </foreach>
</mail>
```

必須 / オプション

オプションです。*foreach* 要素は、必要な数だけ指定できます。

説明

foreach 要素は、繰返しグループに対して配布を一括処理するために使用します。*foreach* を使用できるのは、関連するレポート定義ファイル（RDF、JSP、XML のいずれか）において、一括処理されるセクションに対する "繰返し" プロパティが、該当するグループに設定されている場合に限られます。*foreach* 要素は、開始と終了タグの間に定義されている配布を、各繰返しグループに対して実行するように指定します。

foreach 要素には、次のサブ要素があります。

- [mail](#)
- [file](#)
- [printer](#)
- [destype](#)
- [attach](#)

これらの要素について、次の各項で説明します。

foreach 要素を [mail](#) 要素のサブ要素として使用することもできます。これは、この項の始めにある 2 番目の例に示されているとおりです（この例では、**mainSection** が **G_DEPARTMENT_ID** で繰り返されることを前提として、レポートの **mainSection** のすべてのインスタンスを含んだ 1 つの添付ファイルが 1 つのファイルに作成されます）。

foreach 要素の動作には、[attach](#) 要素および [file](#) 要素の *instance* 属性が密接に関係しています。*foreach* でレコード・グループに従って配布を実行するように指定し、*instance* で、一括処理されるグループを 1 つのファイルで配布するか（*instance*="all"）または別々のファイルとして配布、つまりグループ・インスタンスごとに 1 つのファイルを配布するか（*instance*="this"）を指定します。

mail 要素とともに使用した場合、*foreach* の意味は、*mail* 要素に対するその位置によって変わります。

- *foreach* が *mail* 要素および *instance*="this" の前にある場合は、各グループ・インスタンスが別々のメールとして送信されます。次に例を示します。

```
<foreach>
  <mail id="a1" to="managers@mycompany.com" subject="results">
    <attach name="department_&lt;department_id&gt;.pdf"
      instance="this">
      <include src="mainSection" />
    </attach>
  </mail>
</foreach>
```

レポートが `department_id` に従ってグループ化され、かつ 4 つの部門が存在する場合は、グループ・インスタンスの数が 4 つになります。つまり、受信者ごとに 4 通の電子メールが作成され、各電子メールにその独自のグループ・インスタンスが添付されます。たとえば、ある電子メールに部門 10 のレポート、別の電子メールに部門 20 のレポートが添付され、残りの電子メールにもそれぞれ違う部門のレポートが添付されます。各受信者が、4 通の電子メールすべてを受け取ります。

- *foreach* が `mail` 要素および `instance="this"` の後にある場合は、各グループ・インスタンスが 1 通の電子メールに添付され、それが各受信者に送信されます。次に例を示します。

```
<mail id="a1" to="managers@mycompany.com" subject="results">
  <foreach>
    <attach name="department_&lt;department_id&gt;.pdf"
      instance="this">
      <include src="mainSection" />
    </attach>
  </foreach>
</mail>
```

9.3.3 mail

例

```
<mail id="a1" to="jsmith@foo.com" subject="Results">
  <body srcType="text">
    Attached are quarterly results.
  </body>
  <attach srcType="report">
    <include src="headerSection"/>
    <include src="mainSection"/>
  </attach>
</mail>
```

OR

```
<mail id="a4" to="recipient@mycompany.com" subject="Regional Results">
  <foreach>
    <attach format="pdf" name="report.pdf" srcType="report" instance="this">
      <include src="mainSection"/>
    </attach>
  </foreach>
</mail>
```

必須 / オプション

オプションです。mail 要素は、必要な数だけ指定できます。

説明

mail 要素は、SMTP ベースの発信メール・サーバーを介した配布を指定するために使用します。この要素を使用して、電子メールの受信者、件名および優先度を指定します。

mail 要素には、次の 3 つのサブ要素があります。

- body
- attach
- foreach

mail 要素の開始と終了タグの間には、body サブ要素を 1 つだけ指定できます。また、任意数の attach サブ要素と foreach サブ要素を任意の場所に指定できます。

mail 要素には、関連する属性もあります。これらの属性は、mail タグ内に記述されます。たとえば、id、to、subject の 3 つの属性が記述されます。

```
<mail id="a1" to="jsmith@foo.com" subject="Recent Hires">
```

表 9-1 では、mail 要素に関連する属性を説明します。

表 9-1 mail 要素の属性

属性	有効値	説明
id	文字列	必須。特定の mail 要素を識別する、配布 XML ファイル内で一意のキーワードです。テキスト文字列と 1 つ以上の数値を組み合わせた値を指定できます (id="a1" など)。id 値は、英字で始まる必要があります。
to	文字列	必須。変数値を指定できます。電子メールの受信者を指定します。完全で正式な電子メール・アドレスを含みます (to="jsmith@foo.com" など)。複数の受信者は、カンマで区切ります。 関連レポートで使用されている列を参照する変数値を指定することもできます。詳細は、 9.2.2 項 を参照してください。
cc	文字列	オプション。変数値を指定できます。電子メールのコピーを受け取る受信者を指定します。
bcc	文字列	オプション。変数値を指定できます。電子メールのブラインド・コピーを受け取る受信者を指定します。

表 9-1 mail 要素の属性（続き）

属性	有効値	説明
from	文字列	オプション。変数値を指定できます。電子メールの送信者を指定します。
replyTo	文字列	オプション。変数値を指定できます。返信の送信先となる電子メール・アカウントを指定します。
subject	文字列	デフォルト: Mail Sent from &Report オプション。変数値を指定できます。電子メールの件名を指定します。subject が指定されなかった場合は、件名の行に Mail Sent from [Name of Report] と表示されます。
priority	highest high normal low lowest	デフォルト: normal 電子メールの配信優先度。
returnReceipt	true false	デフォルト: false 電子メールが受信されたときに返信先の個人またはアカウントに通知するかどうかを指定します。
organization	文字列	オプション。変数値を指定できます。電子メールを配布する組織の名前を指定します。次に例を示します。 organization="Region 10 Sales" または organization="&department_name"

注意： mail 要素が正しく機能するには、Reports Server がメールの送信先である発信 SMTP メール・サーバーを認識することが必要です。この情報は、Reports Server 構成ファイル（<server_name>.conf）で指定します。このファイルの pluginParam 要素に、メール・サーバーの名前を入力できます。次に例を示します。

```
<pluginParam name=mailServer>smtp01.mycorp.com</pluginParam>
```

詳細は、第 3 章「Oracle9iAS Reports Services の構成」を参照してください。

9.3.4 body

例

Windows の場合 `<mail id="a1" to="jsmith@foo.com" subject="Results">`
 `<body srcType="file">`
 `<include src="c:¥mail¥body.html"/>`
 `</body>`
`</mail>`

UNIX の場合 `<mail id="a1" to="jsmith@foo.com" subject="Results">`
 `<body srcType="file">`
 `<include src="/mail/body.html"/>`
 `</body>`
`</mail>`

必須 / オプション

オプションです。1 つの mail 要素には、body 要素を 1 つだけ関連付けることができます。

説明

body 要素は、*mail* 要素のサブ要素です。この要素は、電子メールの内容（本文）を指定します。*body* では、*body* タグの開始と終了の間にテキスト文字列を入力するか、または *include* サブ要素を使用して外部ファイル、レポートまたはレポートのセクションを指定できます。次に例を示します。

```
<mail id="a1" to="jsmith@foo.com" subject="Results">
  <body srcType="text">
    Attached are quarterly results.
  </body>
  ...
```

または

```
<mail id="a1" to="jsmith@foo.com" subject="Results">
  <body srcType="file">
    <include src="d:¥reports¥admin¥results.html"/>
  </body>
  ...
```

または

```
<mail id="a1" to="&lt;first_name&gt;. &lt;last_name&gt;@myco.com"
subject="Quarterly Results">
  <body srcType="report" format="html">
    <include src="headerSection"/>
  </body>
  ...
</mail>
```

body には、*srcType*、*format*、*instance* の 3 つの属性があります。表 9-2 では、これらの属性について説明します。

表 9-2 mail の body サブ要素の属性

属性	有効値	説明
srcType	file report text	デフォルト: report 電子メールの内容のソース。内容は電子メールの本文に表示されます。 <i>srcType</i> を指定しなかった場合は、デフォルト値が使用されます。
format	html htmlcss ascii	デフォルト: html <i>srcType</i> が <i>report</i> であり、そのフォーマットがデフォルトの <i>html</i> 以外である場合は必須で、それ以外はオプションです。内容のフォーマットを指定します。
instance	this all	デフォルト: all foreach 要素が指定されている場合に使用します。グループ化されたレポートが別々のレポートに一括処理される場合は、 <i>instance</i> で、グループをグループ・インスタンスごとに個別の内容へと分割するか (<i>this</i>)、またはすべてを同じ内容に収めるか (<i>all</i>) を指定します。

9.3.5 attach

例

```
<mail id="a1" to="jsmith@foo.com" subject="Results">
  <body srcType="text">
    Attached are quarterly results.
  </body>
  <foreach>
    <attach format="html" name="contacts.htm" srcType="report" instance="all">
      <include src="headerSection"/>
      <include src="mainSection"/>
    </attach>
  </foreach>
</mail>
```

必須 / オプション

オプションです。attach 要素は、1 つの mail 要素に対して必要な数だけ指定できます。attach は、foreach のサブ要素でもあることに注意してください。foreach では、そのサブ要素 (mail、file、printer、destype および attach のサブ要素) の少なくとも 1 つを使用する必要があります。

説明

attach は、電子メールの添付ファイルを指定します。attach には少なくとも 1 つの include サブ要素が必要です。srcType="report" である場合は、このサブ要素を複数指定できます。attach の属性は、表 9-3 で説明します。

表 9-3 mail の attach サブ要素の属性

属性	有効値	説明
format	pdf html htmlcss rtf ascii xml dflt	デフォルト: pdf srcType が report であり、レポート・フォーマットがデフォルトの pdf 以外である場合は必須で、それ以外はオプションです。添付ファイルのフォーマットを指定します (format="htmlcss" など)。
name	文字列	オプション。変数値を指定できます。添付ファイルのファイル名を指定します。関連レポートで使用されている列を参照する変数値を指定することもできます。詳細は、9.2.2 項を参照してください。
srcType	file report text	デフォルト: report 添付ファイルのソース (ファイル、レポート、テキストのいずれか)。

表 9-3 mail の attach サブ要素の属性 (続き)

属性	有効値	説明
instance	this all	デフォルト: all <i>foreach</i> 要素が指定されている場合に使用します。グループ化されたレポートが別々のレポートに一括処理される場合は、 <i>instance</i> で、グループをグループ・インスタンスごとに個別の内容へと分割するか (<i>this</i>)、またはすべてを同じ内容に収めるか (<i>all</i>) を指定します。

これらの属性を *foreach* 要素とともに使用することで、グループ・インスタンスで繰り返される宛先を設計し、グループの添付ファイルごとに電子メールを生成できます。次に例を示します。

```
<foreach>
  <mail id="a2" to="first.name@myco.com,second.name@myco.com, third.name@myco.com,
    fourth.name@myco.com" subject="Department Summaries">
    <body srcType="text">
      Attached is the breakdown of department summaries for the last quarter.
    </body>
    <attach format="htmlcss" name="deptsum.html" srcType="report"
      instance="this">
      <include src="report"/>
    </attach>
  </mail>
</foreach>
```

foreach 要素の位置を変えると、複数のファイルが添付された 1 通の電子メールを生成できます。この場合、グループ・インスタンスごとに添付ファイルが異なります。

```
<mail id="a2" to="first.name@myco.com,second.name@myco.com, third.name@myco.com,
  fourth.name@myco.com" subject="Department Summaries">
  <body srcType="text">
    Attached is the breakdown of department summaries for the last quarter.
  </body>
  <foreach>
    <attach format="htmlcss" name="deptsum.html" srcType="report"
      instance="this">
      <include src="report"/>
    </attach>
  </foreach>
</mail>
```

9.3.6 include

例

```
<mail id="a1" to="jsmith@foo.com" subject="Q4">
  <body srcType="text">
    Attached are quarterly results.
  </body>
  <attach srcType="report" format="pdf">
    <include src="report"/>
  </attach>
</mail>
```

または

```
<mail id="a1" to="jsmith@foo.com" subject="Q4">
  <body srcType="text">
    Attached are quarterly results.
  </body>
  <attach srcType="report" format="htmlcss">
    <include src="headerSection"/>
  </attach>
</mail>
```

または

```
<mail id="a1" to="jsmith@foo.com" subject="Q4">
  <body srcType="text">
    Attached are quarterly results.
  </body>
  <attach srcType="file">
    <include src="c:¥management¥reports¥current¥Q4.htm"/>
  </attach>
</mail>
```

必須 / オプション

body および *attach* とともに使用されるときに、*srcType* が *report* または *file* である場合は必須で、*srcType* が *text* である場合はオプションです。*file*、*printer* および *destype* にも必須です。*include* が必要なインスタンスでは、必ず 1 つの *include* を指定する必要があります。複数指定してもかまいません。

説明

include 要素は、*body*、*attach*、*file*、*printer*、*destype* の各要素とともに使用します。この要素は、電子メールの本文に含めるファイル、レポートまたはレポート・セクションを、電子メールの添付ファイル、ファイルの内容、プリンタ出力またはカスタム宛先タイプの内容として指定します。

レポート全体ではなく複数のセクションを指定する場合は、必要なセクションごとに *include* を入力します。次に例を示します。

```
<mail id="a1" to="jsmith@foo.com" subject="Results">
  <body srcType="text">
    Attached are quarterly results.
  </body>
  <attach srcType="report" format="htmlcss">
    <include src="headerSection"/>
    <include src="mainSection"/>
  </attach>
</mail>
```

先に指定された *body* 要素または *attach* 要素の *srcType* 属性が *file* である場合は、これに続く *include* 要素で、そのファイルをディレクトリ・パスとファイル名で指定するか、または単にファイル名で指定できます（そのファイルが `REPORTS_PATH` 環境変数で示されたディレクトリ内にある場合）。次に例を示します。

```
<mail id="a1" to="jsmith@foo.com">
  <body srcType="file">
    <include src="q4sales.pdf"/>
  </body>
</mail>
```

パスを指定する場合は、プラットフォームに適した標準を使用してください。次に例を示します。

Windows の場合： `<include src="c:\management\reports\current\Q4.htm"/>`

UNIX の場合： `<include src="/management/reports/current/Q4.htm"/>`

include 要素の開始タグと終了タグの間に他の XML 要素は記述されません。ただし、*include* には 1 つだけ属性（*src*）があります。表 9-4 では、その属性について説明します。

表 9-4 mail の body または attach とともに使用される include サブ要素の属性

属性	有効値	説明
src	(パスおよび) ファイル名 report headerSection mainSection trailerSection	<p>必須。先行する <i>attach</i>、<i>body</i>、<i>file</i>、<i>printer</i> または <i>destype</i> の各要素に指定されているデータのソースを指定します。</p> <p>配布 XML ファイルは、特定のレポートを実行するとき呼び出されるため、<code>src="report"</code> である場合は、<i>src</i> 属性にレポートの名前や場所を指定する必要があります。</p> <p>先行する <i>body</i> 要素または <i>attach</i> 要素の <i>srcType</i> 属性が <i>file</i> である場合は、ディレクトリ・パスとファイル名を指定するか、または単にファイル名を指定します（そのファイルが <code>REPORTS_PATH</code> 環境変数で示されたディレクトリ内にある場合）。</p> <p>先行する <i>body</i> 要素または <i>attach</i> 要素の <i>srcType</i> 属性が <i>report</i> である場合は、レポート全体を指定するか（<i>report</i>）、あるいは本文に含める（または添付する）レポートのセクションを指定します（<i>headerSection</i>、<i>mainSection</i>、<i>trailerSection</i> のいずれか、またはその組合せ）。</p>

9.3.7 file

例

Windows の場合

```
<file name="c:\management\reports\report.pdf" format="pdf">
  <include src="report"/>
</file>
```

UNIX の場合

```
<file name="/management/reports/report.pdf" format="pdf">
  <include src="report"/>
</file>
```

または

```
<foreach>
  <file name="section&lt;department_id&gt;.pdf" format="pdf" instance="this">
    <include src="mainSection"/>
  </file>
</foreach>
```


必須 / オプション

オプションです。*file* 要素は、必要な数だけ指定できます。

説明

file 要素は、ファイルへの配布を指定するために使用します。*file* 要素には、サブ要素 *include* があります。*file* 要素の開始と終了の間には、少なくとも 1 つの *include* サブ要素が必要です。このサブ要素は複数指定できます。

file 要素を、*foreach* 要素および *instance="this"* 属性とともに使用すると、グループ化されたレポートの各グループ・インスタンスを別々のファイルに配布できます。たとえば、レポートを *department_id* でグループ化し、4 つの部門が存在する場合は、*foreach/file/instance="this"* を組み合わせて、それぞれが別々の部門のレポートを含んだ 4 つのファイルを生成できます。この場合、配布 XML ファイルにおける *file* エントリは、次のようになります。

```
<foreach>
  <file id="a3" name="dept_&lt;&lt;department_id&gt;&gt;.pdf" format="pdf"
    instance="this">
    <include="report"/>
  </file>
</foreach>
```

この例では、すべてのレポート・セクション（ヘッダー、メインおよびトレーラ）を同じグループ・インスタンス（*department_id* など）で繰り返す必要があります。

file 要素には、関連する属性もあります。これらの属性は、*file* タグ内に記述されます。たとえば、*file* の *"id"* 属性と *"name"* 属性は、次のように表されます。

Windows: `<file id="a7" name="d:¥reports¥2001¥q4sales.pdf">`

UNIX: `<file id="a7" name="/reports/2001/q4sales.pdf">`

表 9-5 では、*file* 要素に関連する属性を説明します。

表 9-5 file 要素の属性

属性	有効値	説明
id	文字列	必須。特定の file 要素を識別する、配布 XML ファイル内で一意のキーワードです。テキスト文字列と 1 つ以上の数値を組み合わせた値を指定できます (id="a1" など)。id 値は、英字で始まる必要があります。
name	文字列	必須。変数値を指定できます。宛先ファイルの場所とファイル名を指定します。ディレクトリ・パスを入力します。ファイル名も含めます。次に例を示します。 Windows: name="d:\reports\file.pdf" UNIX: name="reports/file.pdf" 関連レポートで使用されている列を参照する変数値を指定することもできます。詳細は、 9.2.2 項 を参照してください。
format	pdf html htmlcss rtf a scii xml bitmap	デフォルト: pdf 宛先ファイルのフォーマット。次に例を示します。 format="htmlcss"
instance	this all	デフォルト: all <i>foreach</i> 要素が指定されている場合に使用します。グループ化されたレポートが別々のレポートに一括処理される場合は、 <i>instance</i> で、グループをグループ・インスタンスごとに個別のファイルへと分割するか (<i>this</i>)、またはすべてを同じファイルに収めるか (<i>all</i>) を指定します。

9.3.8 printer

例

Windows の場合

```
<printer id="a1" name="¥¥server_name¥printer_name" copies="5">  
  <include src="report"/>  
</printer>
```

UNIX の場合

```
<printer id="a1" name="alias_to_registered_printer" copies="5" instance="all">  
  <include src="report"/>  
</printer>
```

必須 / オプション

オプションです。printer 要素は、必要な数だけ指定できます。

説明

printer 要素は、プリンタへの配布を指定するために使用します。*printer* 要素には、サブ要素 *include* があります。*printer* 要素の開始と終了の間には、少なくとも 1 つの *include* サブ要素が必要です。このサブ要素は複数指定できます。

printer 要素を、*foreach* 要素および *instance="this"* 属性とともに使用すると、グループ化されたレポートの各グループ・インスタンスを別々の印刷ジョブに配布できます。たとえば、レポートを *department_id* でグループ化し、4 つの部門が存在する場合は、*foreach/printer/instance="this"* を組み合わせて、それぞれが別々の部門のレポートを含んだ 4 つの印刷レポートを生成できます。この場合、配布 XML ファイルにおける *printer* エントリは、次のようになります。

```
<foreach>
  <printer id="a7" name="¥¥server_name¥printer_name" instance="this">
    <include="report"/>
  </printer>
</foreach>
```

この例では、すべてのレポート・セクション（ヘッダー、メインおよびトレーラ）を同じグループ・インスタンス（*department_id* など）で繰り返す必要があります。

表 9-6 では、*printer* 要素に関連する属性を説明します。

表 9-6 printer 要素の属性

属性	有効値	説明
id	文字列	必須。特定の <i>printer</i> 要素を識別する、配布 XML ファイル内で一意のキーワードです。テキスト文字列と 1 つ以上の数値を組み合わせた値を指定できます (<i>id="a1"</i> など)。id 値は、英字で始まる必要があります。
name	文字列	必須。変数値を指定できます。宛先のプリンタを指定します。この情報の入力方法は、Windows と UNIX で異なります。 Windows の場合は、プリンタ・サーバー名とプリンタ名を指定します。次に例を示します。 name="¥¥server_name¥printer_name" UNIX の場合は、登録されたプリンタに割り当てられた別名を指定します。次に例を示します。 name="sales_printer" 関連レポートで使用されている列を参照する変数値を指定することもできます。詳細は、9.2.2 項を参照してください。

表 9-6 printer 要素の属性（続き）

属性	有効値	説明
copies	文字列	デフォルト: 1 各レポートまたは各レポート・グループ・インスタンスの印刷部数。
instance	this all	デフォルト: all <i>foreach</i> 要素が指定されている場合に使用します。グループ化されたレポートが別々のレポートに一括処理される場合は、 <i>instance</i> で、グループをグループ・インスタンスごとに別々の印刷レポートに分割するか (<i>this</i>)、またはすべてを同じ印刷レポートに収めるか (<i>all</i>) を指定します。

9.3.9 destype

例

```
<destype id="acustom1" name="fax">  
  <include src="headerSection"/>  
  <property name="number" value="914925551212"/>  
</destype>
```

必須 / オプション

オプションです。destype 要素は、必要な数だけ指定できます。

説明

destype 要素は、FAX や FTP サイトなどのカスタムの宛先への配布を指定するために使用します。また、Oracle9iAS Portal で作成されたポータルへの配布を指定する場合も、destype 要素を使用します。destype 要素では、property および include の 2 つのサブ要素を使用できます。このうち、include は少なくとも 1 つ指定する必要があります。

カスタムの宛先タイプを指定するには、レポートの内容をカスタムの出力先に導くための配布ハンドラを定義する必要があります。

注意： カスタムの宛先タイプは、Oracle9iAS Reports Services Destinations API で作成します。Reports API の最新情報は、Oracle Technology Network (<http://otn.oracle.com/> 英語サイト) または Oracle Technology Network Japan (<http://otn.oracle.co.jp/> 日本語サイト) を参照してください。

destype 要素を、*foreach* 要素および *instance="this"* 属性とともに使用すると、グループ化されたレポートの各グループ・インスタンスを別々の宛先タイプ・インスタンス（別々の FAX など）に配布できます。たとえば、レポートを *department_id* でグループ化し、4 つの部門が存在する場合は、*foreach/printer/instance="this"* を組み合わせて、それぞれが別々の部門のレポートを含んだ 4 つの宛先タイプ・インスタンスを生成できます。この場合、配布 XML ファイルにおける *destype* エントリは、次のようになります。

```
<foreach>
  <destype id="a9" name="fax" instance="this">
    <include="report"/>
    <property name="number" value="&amp;&lt;fax_number&gt;"/>
  </destype>
</foreach>
```

この例では、すべてのレポート・セクション（ヘッダー、メインおよびトレーラ）を同じグループ・インスタンス（*department_id* など）で繰り返す必要があります。

カスタムの宛先タイプには、関連する属性もあります。これらの属性は、*destype* タグ内に記述されます。たとえば、*destype* の "id" 属性、"name" 属性および "instance" 属性は、次のように表されます。

```
<foreach>
  <destype id="a1" name="name_of_destination_type" instance="all">
</foreach>
```

表 9-7 では、*destype* 要素に関連する属性を説明します。

表 9-7 *destype* 要素の属性

属性	有効値	説明
id	文字列	必須。特定の <i>destype</i> 要素を識別する、配布 XML ファイル内で一意のキーワードです。テキスト文字列と 1 つ以上の数値を組み合わせた値を指定できます (<i>id="a1"</i> など)。id 値は、英字で始まる必要があります。
name	文字列	必須。カスタムの宛先の名前を指定します。たとえば、FAX の場合は次のように指定します。 name="fax" Oracle9iAS Portal で作成されたポータルの場合は、次のように指定します。 name="oraclePortal"

表 9-7 destype 要素の属性（続き）

属性	有効値	説明
instance	this all	<p>デフォルト：all</p> <p><i>foreach</i> 要素が指定されている場合に使用します。グループ化されたレポートが別々のレポートに一括処理される場合は、<i>instance</i> で、グループをグループ・インスタンスごとに別々の宛先タイプ・インスタンスに分割するか（<i>this</i>）、またはすべてを同じ宛先タイプ・インスタンスに収めるか（<i>all</i>）を指定します。</p> <p>たとえば、カスタムの宛先タイプが FAX である場合、<i>instance</i>="this" はグループ・インスタンスごとに個別の FAX を使用することを意味し、<i>instance</i>="all" はすべてのグループに同じ FAX を使用することを意味します。</p>

Oracle9iAS Reports Services の環境では、カスタムの宛先タイプ（Pluggable Destinations）の作成と使用がサポートされています。その一環として、配布 XML ファイル内でカスタムの宛先を呼び出すことができます。配布 XML ファイルでは、*destype* 要素とともに使用されるプロパティの名前 / 値のペアを使用して、カスタムの宛先を定義することができます。

9.3.10 property

例

```
<foreach>
  <destype id="custom1" name="fax" instance="all">
    <include src="headerSection"/>
    <property name="number" value="914925551212"/>
  </destype>
</foreach>
```

必須 / オプション

オプションです。*destype* 要素の下には、必要な数のプロパティを指定できます。

説明

property 要素には、カスタムの宛先タイプ（*destype*）が認識できる用語で表された名前 / 値のペアを指定します。プロパティは、単に宛先ハンドラに渡されます。Reports Services 内では特に機能を果たしません。プロパティの指定方法は、使用するカスタムの宛先の要件に完全に依存します。

9.4 配布 XML ファイルの例

この項では、配布 XML 要素の単純な例から複雑な例までを紹介します。これらの例は、次に示す `distribution.dtd` の主要な要素に基づいて編成されています。

- [foreach](#) の例
- [mail](#) の例
- [file](#) の例
- [printer](#) の例

9.4.1 foreach の例

この項で取り上げる例は、次のとおりです。

- [レポート・グループを個別の添付ファイルとする 単一の電子メール](#)
- [グループ・インスタンスごとに個別の電子メール](#)
- [単独のセクションを添付ファイルとする個別の電子メール](#)
- [セクションごとに異なるファイル](#)
- [レポートごとに実行される印刷](#)

9.4.1.1 レポート・グループを個別の添付ファイルとする 単一の電子メール

この例では、各添付ファイルがヘッダー、メイン、トレーラの各セクションの対応するインスタンスを含んでいます。つまり、レポートが `department_id` でグループ化され、最初の部門が部門 10 である場合は、部門 10 の情報を含んでいるヘッダー、メイン、トレーラの各セクションから成るレポートが最初の添付ファイルとなります。この例が有効となるのは、同じグループ・インスタンス（この例の場合は `department_id`）でヘッダー、メイン、トレーラの各セクションが繰り返される場合に限られます。

```
<mail id="a1" to="managers@mycompany.com" subject="New Hires">
  <foreach>
    <attach format="html" srcType="report" instance="this">
      <include src="report"/>
    </attach>
  </foreach>
</mail>
```

まず、この例では、"managers@mycompany.com" が各部門の管理者に配信を行うメーリング・リストのアドレスであることを前提とします。10、20、30、40 の 4 つの部門がある場合は、最初の添付ファイルに部門 10 に対応するヘッダー、メイン、トレーラの各セクションが取り込まれ、後続の各添付ファイルに残りの各部門に対応するセクションが取り込まれます。この例では、4 つの添付ファイルを伴う電子メールが受信者ごとに 1 通ずつ作成されます。

9.4.1.2 グループ・インスタンスごとに個別の電子メール

この例では、各受信者が、グループ化されたレポートごとに異なる電子メールを受け取ります。たとえば、レポートが department_id でグループ化され、4 つの部門が存在する場合は、1 人の受信者がそれぞれに各部門のレポートが添付された 4 通の電子メールを受け取ります。

```
<foreach>
  <mail id="weeklies" to="managers@mycompany.com">
    <attach format="htmlcss" srcType="report" instance="this">
      <include src="mainSection"/>
    </attach>
  </mail>
</foreach>
```

9.4.1.3 単独のセクションを添付ファイルとする個別の電子メール

この例では、個別のセクションが別々のグループで繰り返されます。各受信者が、グループ化されたメイン・セクションごと、およびグループ化されたトレーラ・セクションごとの添付ファイルを伴う個別の電子メールを受信するように配布が設定されます。

```
<foreach>
  <mail id="a6" to="managers@mycompany.com" subject="Personnel Reports">
    <attach format="pdf" name="attach.pdf" srcType="report" instance="this">
      <include src="mainSection"/>
    </attach>
    <attach format="rtf" name="attach.rtf" srcType="report" instance="this">
      <include src="trailerSection"/>
    </attach>
  </mail>
</foreach>
```


9.4.1.4 セクションごとに異なるファイル

この例では、グループ・インスタンスごとに異なるファイルが生成されます。グループは `department_id` で繰り返されます。各ファイルには、関連する部門 ID から名前が付けられます。

```
<foreach>
  <file id="a10" name="department_&lt;&lt;department_id&gt;&gt;.pdf" instance="this">
    <include src="mainSection"/>
  </file>
</foreach>
```

この例では、10 から 40 までの 4 つの部門があることを前提に、`department_10.pdf`、`department_20.pdf` などの名前で 4 つのファイルが作成されます。

`file` 要素には `format` 属性が含まれていません。これは、`srcType` が `file` または `text` である場合は、この属性が不要であるためです。`srcType` が `report` である場合は、この属性が必要となります。

注意： この例で、変数値を使用するために一意のファイル名を指定しない場合（9.2.2 項を参照）は、後から作成される各ファイルによって、前に作成されたファイルが上書きされます。つまり、部門 20 の `department.pdf` ファイルによって部門 10 の `department.pdf` ファイルが上書きされます。この上書きは、最後の部門（部門 40）レポートの情報を含んだ `department.pdf` が唯一のファイルとなるまで続きます。

9.4.1.5 レポートごとに実行される印刷

プリンタ名の指定方法は、Windows と UNIX で異なります。最初に Windows の例を示します。次に UNIX の例を示します。

9.4.1.5.1 Windows この例では、レポートが `department_id` でグループ化されるという前提で、レポートが部門ごとに印刷されます。

```
<foreach>
  <printer id="a7" name="¥¥server_name¥printer_name" instance="this">
    <include src="report"/>
  </printer>
</foreach>
```

9.4.1.5.2 UNIX この例では、レポートが `department_id` でグループ化されるという前提で、レポートが部門ごとに印刷されます。

```
<foreach>
  <printer id="a7" name="printer_alias" instance="this">
    <include src="report"/>
  </printer>
</foreach>
```

9.4.2 mail の例

この項で取り上げる例は、次のとおりです。

- レポート全体を本文とする電子メール
- レポートの 1 つのセクションを本文とする電子メール
- レポートの 2 つのセクションを本文とする電子メール
- 外部ファイルを本文とし、レポートを添付ファイルとする 電子メール
- レポート全体およびグループ化されたセクションが添付された 電子メール
- 関連する管理者と部門に送信される電子メール

9.4.2.1 レポート全体を本文とする電子メール

レポートがこの電子メールの内容となります。つまり、受信者がこの電子メールを開くと、レポートが表示されます。

```
<mail id="a5" to="managers@mycompany.com" subject="Quarterly Report">
  <body srcType="report" format="html">
    <include src="report"/>
  </body>
</mail>
```

9.4.2.2 レポートの 1 つのセクションを本文とする電子メール

レポートの 1 つのセクションがこの電子メールの内容となります。つまり、受信者がこの電子メールを開くと、レポートのセクションが表示されます。

```
<mail id="a6" to="employees@mycompany.com">
  <body srcType="report" format="html">
    <include src="mainSection"/>
  </body>
</mail>
```

この `mail` 要素には `subject` 属性が含まれないため、デフォルトの件名である `Mail Sent From &Report` が使用されます。変数 `&Report` は、実行時にレポートの名前で置き換えられます。

9.4.2.3 レポートの2つのセクションを本文とする電子メール

レポートの2つのセクションがこの電子メールの内容となります。つまり、受信者がこの電子メールを開くと、1つのレポートに結合された2つのセクション（headerSection と mainSection）が表示されます。

```
<mail id="emp_addresses" to="employees@mycompany.com" subject="Employee Address
List">
  <body srcType="report" format="html">
    <include src="headerSection"/>
    <include src="mainSection"/>
  </body>
</mail>
```

9.4.2.4 外部ファイルを本文とし、レポートを添付ファイルとする電子メール

このメールでは、外部ファイルが本文の内容となり、レポートが添付ファイルとなります。外部ファイルへのパスの表し方は、Windows と UNIX で異なります。

9.4.2.4.1 Windows

```
<mail id="XQRSN" to="accounting@mycompany.com" subject="Salaries"
  <body srcType="file">
    <include src="c:¥mail¥body.html"/>
  </body>
  <attach format="pdf" name="salaries.pdf" srcType="report">
    <include src="report"/>
  </attach>
</mail>
```

9.4.2.4.2 UNIX

```
<mail id="XQRSN" to="accounting@mycompany.com" subject="Salaries"
  <body srcType="file">
    <include src="/mail/body.html"/>
  </body>
  <attach format="pdf" name="salaries.pdf" srcType="report">
    <include src="report"/>
  </attach>
</mail>
```

9.4.2.5 レポート全体およびグループ化されたセクションが添付された電子メール

この例では、受信者が複数の添付ファイルを伴う 1 通の電子メールを受け取ります。その内訳は、グループ・インスタンスごとの 1 つの添付ファイルと、レポート全体を含んだ 1 つの添付ファイルです。レポートが `department_id` でグループ化され、4 つの部門が存在する場合は、受信者が 5 つの添付ファイルを受け取ります（部門ごとに 1 つの添付ファイルとレポート全体の添付ファイル）。

```
<mail id="grx90" to="sales@mycompany.com">
  <body srcType="text">
    Attached you will find the summary report and breakdown by department of weekly
    totals.
  </body>
  <attach format="rtf" name="myAttach.rtf" srcType="report">
    <include src="report"/>
  </attach>
  <foreach>
    <attach format="pdf" name="myattach.pdf" srcType="report" instance="this">
      <include src="mainSection"/>
    </attach>
  </foreach>
</mail>
```

9.4.2.6 関連する管理者と部門に送信される電子メール

この例では、部門 10 の管理者が部門 10 のレポートを受け取り、部門 20 の管理者が部門 20 のレポートを受け取ります。このタグを有効にするには、変数の参照先が、添付されるセクションとともに使用される "繰り返し" グループに含まれている列である必要があります。つまり、セクションが `G_department_id` で繰り返される場合は、*manager* がそのグループの列である必要があります。

```
<foreach>
  <mail id="mgr1090" to="&lt;manager>@mycompany.com">
    <attach format="pdf" name="attach.pdf" srcType="report" instance="this">
      <include src="mainSection"/>
    </attach>
  </mail>
</foreach>
```

9.4.3 file の例

グループ化されたレポートを一括処理してファイルに配布する場合は、繰返しグループに基づいた変数値またはその他の変数情報を使用してファイル名を指定する必要があります。そうしないと、ファイルが作成されるたびに、前に作成されたファイルが上書きされるおそれがあります。たとえば、`department.pdf` の出力ファイル名を指定し、各部門のレポートのインスタンスを個別に出力すると、2 番目の `department.pdf` ファイルによって最初の `department.pdf` ファイルが上書きされ、3 番目の `department.pdf` ファイルによって 2 番目の `department.pdf` ファイルが上書きされます。その結果、最後に出力された部門のレポートのみが残ることになります。グループ化されたレポートをグループ・インスタンスごとに個別に出力する場合は、ファイル名を指定する変数値を使用します。たとえば、`name="department_<department_id>.pdf"` と指定します。

この項で取り上げる例は、次のとおりです。

- レポート全体のファイル
- 結合されたレポート・セクションのファイル
- 結合されたセクションから成る各グループのファイル
- レポート・グループ・インスタンスごとのファイル

9.4.3.1 レポート全体のファイル

この例では、レポート全体を含む `report.pdf` というファイルが作成されます。

9.4.3.1.1 Windows

```
<file id="a1" name="c:\reports\report.pdf" format="pdf">
  <include src="report"/>
</file>
```

9.4.3.1.2 UNIX

```
<file id="a1" name="/reports/report.pdf" format="pdf">
  <include src="report"/>
</file>
```

9.4.3.2 結合されたレポート・セクションのファイル

この例では、レポートのヘッダー・セクションとメイン・セクションで構成されるレポートを含んだ、`sections.pdf` というファイルが作成されます。

```
<file id="a2" name="sections.pdf" format="pdf">
  <include="headerSection"/>
  <include="mainSection"/>
</file>
```

9.4.3.3 結合されたセクションから成る各グループのファイル

この例では、繰り返しグループごとに異なるファイルが作成されます。各ファイルには、関連するグループのメイン・セクションとトレーラ・セクションを結合したレポートが格納されます。メイン・セクションとトレーラ・セクションが、同じグループで繰り返される必要があります。また、変数のファイル名が、" 繰り返し " グループ内の列を参照する必要があります。つまり、レポートが `department_id` で繰り返され、10 から 40 までの 4 つの部門が存在する場合は、あるファイルに部門 10 のメイン・セクションとトレーラ・セクションが収められ、次のファイルに部門 20 のメイン・セクションとトレーラ・セクションが収められ、残りの部門も同様に処理されます。つまり、`name` の中の変数値が、`G_department_id` グループに含まれている列を参照する必要があります。

```
<foreach>
  <file id="file9" name="department_&&lt;department_id&>.pdf"
    instance="this">
    <include src="mainSection"/>
    <include src="trailerSection"/>
  </file>
</foreach>
```

9.4.3.4 レポート・グループ・インスタンスごとのファイル

この例では、レポートが `department_id` でグループ化され、10 から 40 までの 4 つの部門が存在することを前提として、4 つのファイルを作成します。それぞれのファイル名は、`department_10.pdf`、`department_20.pdf`、`department_30.pdf` および `department_40.pdf` です。

```
<foreach>
  <file id="a20" name="department_&&lt;department_id&>.pdf" instance="this">
    <include src="report"/>
  </file>
</foreach>
```

9.4.4 printer の例

この項で取り上げる例は、次のとおりです。

- レポート全体の印刷
- レポートの 2 つのセクションの印刷
- グループ化されたレポートの印刷
- 結合されたセクションをグループ・インスタンスごとに印刷
- レポートの適切なインスタンスをその関連プリンタに印刷

プリンタ名の指定方法は、Windows と UNIX で異なります。各例では、それぞれの指定方法を示します。

9.4.4.1 レポート全体の印刷

この例では、レポート全体が指定のプリンタに送信されます。

9.4.4.1.1 Windows

```
<printer id="a80" name="¥¥neptune¥prtr20">
  <include src="report"/>
</printer>
```

9.4.4.1.2 UNIX

```
<printer id="a80" name="10th_floor_printer">
  <include src="report"/>
</printer>
```

9.4.4.2 レポートの2つのセクションの印刷

この例では、レポートの2つのセクションがプリンタに送信されます。

9.4.4.2.1 Windows

```
<printer id="a1" name="¥¥neptune¥prtr20">
  <include src="headerSection"/>
  <include src="mainSection"/>
</printer>
```

9.4.4.2.2 UNIX

```
<printer id="a1" name="10th_floor_printer">
  <include src="headerSection"/>
  <include src="mainSection"/>
</printer>
```

9.4.4.3 グループ化されたレポートの印刷

この例では、1つのレポートが印刷されます。このレポートは、`department_id`などでグループ化されます。そのためには、レポートのすべてのセクションを同じグループで繰り返す必要があります。

9.4.4.3.1 Windows

```
<foreach>
  <printer id="prt20" name="¥¥neptune¥prtr20" instance="all">
    <include src="report"/>
  </printer>
</foreach>
```

9.4.4.3.2 UNIX

```
<foreach>
  <printer id="prt20" name="10th_floor_printer" instance="all">
    <include src="report"/>
  </printer>
</foreach>
```

9.4.4.4 結合されたセクションをグループ・インスタンスごとに印刷

この例では、グループ・インスタンスごとに1つずつ印刷ジョブが生成されます。結合されたセクションを、同じグループで繰り返す必要があります。レポートが department_id で繰り返され、10 から 40 までの4つの部門が存在する場合は、4つの印刷ジョブが生成されます。つまり、部門10に1つのジョブが生成され、残りの部門にも1つずつジョブが生成されます。メイン・セクションとトレーラ・セクションの両方が、department_id で繰り返される必要があります。

9.4.4.4.1 Windows

```
<foreach>
  <printer id="prt20" name="¥¥neptune¥prtr20" instance="this">
    <include src="mainSection"/>
    <include src="trailerSection"/>
  </printer>
</foreach>
```

9.4.4.4.2 UNIX

```
<foreach>
  <printer id="prt20" name="10th_floor_printer" instance="this">
    <include src="mainSection"/>
    <include src="trailerSection"/>
  </printer>
</foreach>
```


9.4.4.5 レポートの適切なインスタンスをその関連プリンタに印刷

この例が機能するには、"繰返し" グループにホスト・プラットフォームの適切なプリンタ名を含んだ列が含まれている必要があります。つまり、`printer_name` 列に適切なプリンタの別名 (UNIX) またはプリンタ・サーバー / 名前の組合せ (Windows) が含まれている必要があります。たとえば、レポートが `department_id` でグループ化される場合は、`G_department_id` に `printer_name` 列が必要となります。`printer_name` が部門に関連付けられていると仮定して、部門 10 のレポートが部門 10 のプリンタに印刷され、同様に残りの部門のレポートがその部門のプリンタに印刷されます。

```
<foreach>
  <printer id="a60" name="&printer_name" instance="this">
    <include src="mainSection"/>
  </printer>
</foreach>
```

各グループ・インスタンスは、個別の印刷ジョブに相当します。各印刷ジョブは、関連する部門のプリンタに送信されます。

9.5 実行時の配布 XML ファイルの使用方法

実行時に配布 XML ファイルを使用する方法は、URL またはコマンドラインのどちらで使用する場合も基本的に同じです。次のコマンドを使用します。

```
distribute=yes destination=filename.xml
```

filename は、配布 XML ファイルの名前です。XML ファイルの相対パスまたは絶対パスを指定する必要があります。たとえば、Windows の場合は次のように指定します。

```
distribute=yes destination=c:\ORACLE_HOME\reports\distribution\filename.xml
```

UNIX の場合は次のように指定します。

```
distribute=yes destination=$ORACLE_HOME/reports/distribution/filename.xml
```

例に示されているパスは、一例にすぎません。配布 XML ファイルの格納場所には制約がありません。必要な場所に保存することができます。

注意： コマンドラインおよび URL からレポートを実行する操作、および `cgicmd.dat` ファイルの使用に関する詳細は、第 8 章「レポート要求の実行」を参照してください。

9.6 カスタム / Pluggable Destinations に対する XSL 変換

distribution.xml ファイルは、XML スタイル・シートであり、Windows と UNIX どちらの場合も、`ORACLE_HOME\reports\conf\distribution.xml` にあります。このファイルを修正するには、`destype` タグのフォーマットを `distribution.dtd` ファイルに定義されている必要なフォーマットへ変換するためのテンプレートを追加します。

distribution.xml ファイルは、配布 XML ファイル内のユーザー定義カスタム・タグを、Reports Runtime で必要なフォーマットに変換するために使用します。Reports が認識できるのは、すべての Pluggable Destinations に使用される汎用の `destype` タグ構造だけです。ユーザーは、この汎用の `destype` タグ構造に従ってカスタムの宛先を指定できます。これとは別に、使いやすさを考慮して、より具体的なカスタムのタグ構造を指定することもできます。distribution.dtd はこのようなタグを認識できないため、distribution.dtd に指定されているような汎用の `destype` タグ構造にこれらのタグを対応付ける必要があります。

次の例では、FAX が宛先になっています。ユーザーは、汎用のタグ構造に従って、次のように配布 XML ファイルに宛先を指定できます。

```
<destype id="faxdest" name="fax">
  <property name="number" value="123456789"/>
  <include src="report"/>
</destype>
```

使いやすい例：

```
<fax id="faxdest" number="123456789">
  <include src="report"/>
</fax>
```

Reports Runtime は、ここに示した `<fax>` タグ構造を処理できません。これは、`<fax>` タグが `distribution.dtd` ファイルに指定される標準の宛先ではないためです。したがって、このタグ構造を、最初の例に示した汎用フォーマットに変換する必要があります。

そのためには、FAX 宛先用のテンプレートを `distribution.xml` ファイルに指定する必要があります。このテンプレートを使用して、`<fax>` タグ構造が汎用の `destype` 構造に変換されます。distribution.xml のエントリは次のようになります。

```
<xsl:output doctype-system="distribution.dtd"/>

<xsl:template match = "/">
  <xsl:apply-templates match = "destinations" />
</xsl:template>

<xsl:template match="destinations">
  <destinations>

    <!--
    The Standard mail/file/printer/destype and foreach must be copied to the
```

```
transformed xml. The foreach tag must be copied only if it is specified with
file/mail/printer/destype tags.
-->
    <xsl:copy-of select="mail"/>
    <xsl:copy-of select="file"/>
    <xsl:copy-of select="printer"/>
    <xsl:copy-of select="destype"/>
    <xsl:copy-of select="foreach"/>

    <!-- apply template for the sample FAX destination -->
    <xsl:apply-templates match = "fax" />

</destinations>
</xsl:template>

<!--
    Sample Transformation Template for a FAX destination specified in the
    distribution.xml file

    <fax id="FAXDEST" number="123456789">
        <include src="report"/>
    </fax>
-->

<xsl:template match="fax">
    <!-- create a new destype element -->
    <xsl:element name="destype">
        <!--
            create an ID attribute and copy the value from the ID given for the fax
            destination
        -->
        <xsl:attribute name="id">
            <xsl:value-of select="@id"/>
        </xsl:attribute>

        <!-- create a Name attribute with a fax as it's value -->
        <xsl:attribute name="name">fax</xsl:attribute>
        <!--
            create a Property Attribute with name / value attribute pairs property tag is
            created for number attribute. Similarly create more property tags for any other
            attribute you add to the FAX destination
        -->
        <xsl:element name="property">
            <xsl:attribute name="name">number</xsl:attribute>
            <xsl:attribute name="value">
                <xsl:value-of select="@number"/>
            </xsl:attribute>
```

```
</xsl:element>

    <!-- copy the include tag as it is -->
    <xsl:copy-of select="include"/>
</xsl:element>
<!-- end of template -->
</xsl:template>

</xsl:stylesheet>
```

注意： XSL ファイルを修正した後は、そのファイルを同じ場所に同じファイル名で保存します。Reports は、宛先を解決する際に自動的にこの XSL ファイルを検索します。

XML によるレポートのカスタマイズ

XML カスタマイズを使用すると、実行時に元のレポートを変更することなくレポートをカスタマイズできます。ランタイム・コマンドラインに `CUSTOMIZE` コマンドを追加し、カスタマイズ・ファイルを呼び出して、レポートのレイアウトやデータ・モデルへの追加や変更を実行できます。1 つの XML カスタマイズ・ファイルで、これらすべての作業または任意の作業の組合せを実行できます。Reports XML を使用して、カスタムの JSP ベース・レポートに取り込むレポート・データ・モデルを作成することも可能です。

様々な XML カスタマイズを作成して適用すると、レポート出力をユーザー単位またはユーザー・グループ単位で変更できます。同じレポートを使用して、利用対象者ごとに異なる出力を生成できます。

XML カスタマイズをレポートに適用するときに、結合した定義をファイルに保存することができます。このため、XML カスタマイズを使用して、既存のレポートをバッチ更新できます。Reports Builder で 1 つずつファイルを開くことなく、大量のレポートをすばやく更新できます。

Oracle9iAS Reports Services では、Reports XML によるカスタマイズのタイプを拡大し、XML でレポート・データ・モデル全体を作成できるようになりました。これには、複数データ・ソースの作成、データ・ソース間のリンク、および各データ・ソース内でのグループ階層が含まれます。Reports XML カスタマイズによるデータ・モデルのサポートによって、Reports Builder で作成可能なすべてのデータ・モデルを、XML を指定して作成できるようになりました。さらに、データ・モデル・オブジェクトに設定可能なすべてのプロパティを、XML で設定できるようになりました。

この章では、XML を使用して実行中にレポートをカスタマイズする方法、および XML でデータ・モデルを作成する方法について説明します。この章には、次の項があります。

- [カスタマイズの概要](#)
- [XML カスタマイズの作成](#)
- [XML データ・モデルの作成](#)
- [実行時の XML ファイルの使用方法](#)
- [XML レポート定義のデバッグ](#)

この章では、reports.dtd ファイルに含まれる要素の一部のみを紹介し、その例を示します。reports.dtd は、Reports XML に関連するすべての要素と属性を含むデータ型定義ファイルです。要素と属性について、この章の説明よりもさらに詳細な情報が必要な場合は、次の 3 つのソースを参照してください。

- reports.dtd ファイルには、Reports XML のすべての要素と属性、および属性のデフォルト値（存在する場合）が含まれます。reports.dtd ファイルは、Windows と UNIX どちらの場合も、`%ORACLE_HOME%\reports\reports\reports.dtd` にあります。サブ要素の多くは、使用時の規則を示す記号を含んでいます。次に例を示します。
 - プラス記号 (+) は、XML ファイルでそのタイプの要素を 1 つ以上使用できることを意味します。
 - アスタリスク (*) は、XML ファイルでそのタイプの要素を 0 ～多数の範囲で使用できることを意味します。
 - 疑問符 (?) は、XML ファイルでそのタイプの要素を 0 または 1 つだけ使用できることを意味します。
 - 記号が付いていない場合は、XML ファイルでそのタイプの要素を 1 つだけ使用する必要がある（必須である）ことを意味します。

複数のサブ要素が括弧で囲まれ、その直後に記号が付いている場合は、その記号が括弧内のすべてのサブ要素に適用されます。

- Report Builder のオンライン・ヘルプでは、関連するプロパティまたはオブジェクトを検索できます。たとえば、XML の属性 *width* については、Report Builder のオンライン・ヘルプで Width プロパティを検索します。属性の一部は、レポートの内部プロパティから取得されます。内部プロパティは読み込み専用であり、変更できません。オンライン・ヘルプでは、内部プロパティの説明はありません。
- 作成しようとするカスタマイズのタイプを含んだレポートを作成し、レポートを XML として保存し、さらにテキスト・エディタで保存したファイルを表示します。これは、実際の Reports XML を参照する優れた手段であると同時に、より複雑なモデルを構築する場合の参考になります。

10.1 カスタマイズの概要

Reports Builder で作成できるすべてのものは、Reports XML のタグでも作成できます。このため、処理の対象として可能なすべてのレポート・オブジェクトに対してカスタマイズを実行できます。

注意： Reports XML のタグを使用してレポート全体を手動で作成することも可能ですが、ドキュメントに記載され、サポートされているのは、手動で作成されるカスタマイズおよびデータ・モデルのみです。

XML カスタマイズの作成と適用は、3 つのステップで行われます。

1. Reports XML のタグを使用してカスタマイズ・ファイルを作成します。

このカスタマイズは、Oracle9iAS Reports Builder でレポートを作成し、そのレポートを XML として保存することで作成できます。任意のテキスト・エディタまたは高機能の XML エディタを使用して、手動でカスタマイズを作成することもできます。これは、特別なレポート・カスタマイズに必要な XML タグを使用する場合に限られます。

2. XML カスタマイズを、Oracle9iAS Reports Services でアクセス可能な場所に格納します。
3. XML カスタマイズを別のレポートに適用します（CUSTOMIZE コマンドライン引数または PL/SQL のビルトイン `SRW.APPLY_DEFINITION` を使用）。カスタマイズに完全なレポート定義が含まれている場合は、XML カスタマイズを単独で実行できます（REPORT または MODULE コマンドライン引数を使用）。

注意： これらのコマンドライン引数の使用方法の詳細は、付録 A「コマンドライン引数」を参照してください。

10.2 XML カスタマイズの作成

この項では、様々なレポート・カスタマイズの例を示します。ここで取り上げる例は、次のとおりです。

- 必須 XML タグ
- スタイルの変更
- 書式マスクの変更
- 書式例外の追加
- プログラム・ユニットとハイパーリンクの追加
- 新規問合せの追加と新規ヘッダー・セクションでの結果の使用

10.2.1 必須 XML タグ

すべての XML カスタマイズには、次のタグ・ペアが必要です。

```
<report></report>
```

たとえば、最小限の XML カスタマイズは次のとおりです。

```
<report name="emp" DTDVersion="9.0.2.0.0">
</report>
```

この XML カスタマイズには何も含まれていないので、レポートに適用しても効果は一切ありません。必要なタグがあるので解析はできますが、単に必須タグの例として示しています。

`<report>` タグは、レポート・カスタマイズの開始、レポート・カスタマイズ名およびこの XML カスタマイズで使用するデータ型ディクショナリ (DTD) ファイルのバージョンを示します。`</report>` タグは、レポート・カスタマイズの終了を示します。

`report` 要素の `name` 属性には、任意の名前を指定できます。たとえば、XML ファイルでカスタマイズするレポートの名前でも、それ以外の名前でもかまいません。

この例では、`<report>` タグが最小限で使用されています。`<report>` タグにも多数の属性がありますが、そのほとんどは暗黙的に示されており、指定する必要はありません。

`<report>` に唯一必要な属性は、`DTDVersion` です。

完全なレポート定義には、データ・モデルとレイアウトの両方が必要です。したがって、次のタグとその内容を定義する必要があります。

- `<data></data>`
- `<layout></layout>`

`data` 要素には、属性がありません。`layout` 要素には、`panelPrintOrder` と `direction` の 2 つの属性があり、その両方が必須です。これらの属性にデフォルト値 (それぞれ `acrossDown` と `default`) を使用する場合は、属性を指定する必要がありません。`data` 要素と `layout` 要素の例は、次の各項で示します。

10.2.2 スタイルの変更

この項の例では、XML を使用して、レポート・フィールド `F_Mincurrent_pricePersymbol` および `F_Maxcurrent_pricePersymbol` に使用されている塗りと線の色を変更しています。

```
<report name="anyName" DTDVersion="9.0.2.0.0">
  <layout>
    <section name="main">
      <field name="F_Mincurrent_pricePersymbol"
        source="Mincurrent_pricePersymbol"
        lineColor="black"
```



```

        fillColor="r100g50b50"/>
        <field name="F_Maxcurrent_pricePersymbol"
        source="Maxcurrent_pricePersymbol"
        lineColor="black"
        fillColor="r100g50b50"/>
    </section>
</layout>
</report>

```

この例では、*section* 要素と *field* 要素の **name** 属性が、この XML ファイルでカスタマイズされるレポートのメイン・セクションに含まれるフィールドの名前と一致していることが前提となっています。この前提を踏まえて、*field* 要素のその他の属性は、レポートのメイン・セクション内にある同じ名前のフィールドのみに適用されます。

10.2.3 書式マスクの変更

この項の例では、XML を使用して、レポート・フィールド `f_trade_date` に使用される書式マスクを変更しています。

```

<report name="anyName" DTDVersion="9.0.2.0.0">
    <layout>
        <section name="main"
        <field name="f_trade_date"
        source="trade_date"
        formatMask="MM/DD/RR"/>
        </section>
    </layout>
</report>

```

field 要素が独自の終了書式 (`/>`) を使用していることに注意してください。*field* 要素が追加のサブ要素を使用する場合は、*field* 要素を `</field>` で閉じます。

10.2.4 書式例外の追加

この項の例では、XML を使用して、レポートの `f_p_e` および `f_p_e1` フィールドで 10 より大きな値をハイライトする書式例外を追加しています。

```

<report name="anyName" DTDVersion="9.0.2.0.0">
    <layout>
        <section name="main">
            <field name="f_p_e" source="p_e">
                <exception textColor="red">
                    <condition source="p_e" operator="gt" operand1="10"/>
                </exception>
            </field>
            <field name="f_p_e1" source="p_e">
                <exception textColor="blue">

```

```

        <condition source="p_e" operator="gt" operand1="10"/>
      </exception>
    </field>
  </section>
</layout>
</report>
```

この例では、*operator* の値が、" 大なり " を示す *gt* に設定されています。*operator* には、[表 10-1](#) に示すものを指定できます。

表 10-1 operator 属性の値

演算子	使用方法
eq	等しい
lt	より小さい
lteq	以下
neq	等しくない
gt	より大きい
gteq	以上
btw	範囲内
notBtw	範囲外
like	類似
notLike	非類似
null	ヌル
notNull	非ヌル

前の例とは異なり、この例の *field* 要素は、サブ要素を使用しているために、自己完結型の終了書式 (*/>*) ではなく、*</field>* で閉じられていることにも注意してください。

10.2.5 プログラム・ユニットとハイパーリンクの追加

この項の例では、XML を使用してレポートにプログラム・ユニットを追加しています。このプログラム・ユニットによって、従業員の社会保障番号 (SSN) から従業員の詳細にアクセスするハイパーリンクが追加されます。

```
<report name="anyName" DTDVersion="9.0.2.0.0">
  <layout>
    <section name="header">
      <field name="F_ssn1"
        source="ssn1"
```

```

        formatTrigger="F_ssnFormatTrigger"/>
    </section>
    <section name="main">
        <field name="F_ssn"
            source="ssn"
            formatTrigger="F_ssnFormatTrigger"/>
    </section>
</layout>
<programUnits>
    <function name="F_ssnFormatTrigger">
        <textSource>
            <![CDATA[
                function F_ssnFormatTrigger return boolean is
                begin
                    SRW.SET_HYPERLINK('#EMP_DETAILS_&' || LTRIM(TO_CHAR(:SSN))
                    || '>');
                    return (TRUE);
                end;
            ]]>
        </textSource>
    </function>
    <function name="F_ssnFormatTrigger">
        <textSource>
            <![CDATA[
                function F_ssnFormatTrigger return boolean is
                begin
                    SRW.SET_LINKTAG('EMP_DETAILS_&' || LTRIM(TO_CHAR(:SSN)) ||
                    '>');
                    return (TRUE);
                end;
            ]]>
        </textSource>
    </function>
</programUnits>
</report>

```

CDDATA タグは、PL/SQL を XML と区別する目的で、PL/SQL の周辺で使用されています。XML ファイルに HTML を埋め込むときも、同じタグ・シーケンスを使用します。この例では、*field* 要素の *formatTrigger* 属性の名前でファンクションが参照されています。

10.2.6 新規問合せの追加と新規ヘッダー・セクションでの結果の使用

この項の例では、XML を使用して、レポートに新しい問合せを追加するとともに、その問合せの結果を利用する新しいヘッダー・セクションを追加しています。

```

<report name="ref" DTDVersion="9.0.2.0.0">
    <data>

```

```
<dataSource name="Q_summary">
  <select>select portid ports, locname locations from portdesc
</select>
</dataSource>
</data>
<layout>
  <section name="header">
    <tabular name="M_summary" template="BLAFbeige.tdf">
      <labelAttribute font="Arial"
        fontSize="10"
        fontStyle="bold"
        textColor="white"/>
      <field name="F_ports"
        source="ports"
        label="Port IDs"
        font="Arial"
        fontSize="10"/>
      <field name="F_locations"
        source="locations"
        label="Port Names"
        font="Arial"
        fontSize="10"/>
    </tabular>
  </section>
</layout>
</report>
```

この例の XML は、データ・モデルと完全なレイアウトを備えているため、単独で実行できます。

列名が必ず一意となるように、SELECT 文では別名を使用します。別名を使用しなかった場合は、レポート列のデフォルト名が使用されるため、予定外の名前になる可能性があります（たとえば、**portid** ではなく **portid1** など）。*field* 要素の *source* 属性を指定する場合は、ソース列（フィールド）の正しい名前を使用する必要がありますので、この名前の指定は重要なポイントになります。

labelAttribute 要素は、レイアウト内のフィールド・ラベルの書式を定義します。これは *field* 要素の開始と終了タグの外側にあるため、表レイアウトのすべてのラベルに適用されます。この書式をいずれか 1 つのフィールドに関連付ける場合は、この要素を `<field></field>` タグ・ペアの内側に配置します。グローバルおよびローカルの *labelAttribute* 要素が両方とも（1 つは `<field></field>` タグ・ペアの外側に、もう 1 つは内側に）存在する場合は、ローカルなラベル属性がグローバルな属性より優先されます。

10.3 XML データ・モデルの作成

Oracle9iAS Reports Services では、Reports XML のタグを使用して、より高度なデータ・モデルのタイプを作成できるようになりました。次の処理に Reports XML を使用できます。

- 複数のデータ・ソースの作成
- データ・ソース間のリンク
- 各データ・ソース内でのグループ階層の作成
- クロス積（マトリックス）グループの作成
- あらゆるレベルでの式、サマリーおよびプレースホルダの作成
- パラメータの作成

この項では、Reports XML を使用したこれらの処理の例を示します。

これらのデータ・モデル・タイプに加えて、Oracle9iAS Reports Services では Reports XML での PL/SQL の使用をサポートしています。これには、ローカルなプログラム・ユニット、レポートレベルのトリガーおよび連結 PL/SQL ライブラリに対するサポートが含まれます。

10.3.1 複数のデータ・ソースの作成

`<data>` タグが、新しいトランスポータブルなデータ・ソースと複数のデータ・ソースの作成をサポートするようになりました。各データ・ソースは、それぞれの `<dataSource>` タグ内に記述されます。`dataSource` 要素のデータ型定義は、次のとおりです。

```
<!ELEMENT dataSource
  ((select|plugin|plsql),
   comment?,
   displayInfo?,
   formula*,
   group*)>
<!ATTLIST dataSource
  name CDATA #IMPLIED
  defaultGroupName CDATA #IMPLIED
  maximumRowsToFetch CDATA #IMPLIED>
```

次の例では、2 つの SQL データ・ソースを作成し、それぞれに `Q_1` および `Q_2` という名前を付けます。また、各データ・ソースに必要なすべての列を作成するとともに、デフォルト・グループを作成します。このグループの名前には、指定の `defaultGroupName` が設定されます。`defaultGroupName` が指定されていない場合は、デフォルトで独自の名前が設定されます。

```
<report name="anyname" DTDVersion="9.0.2.0.0">
  <data>
    <dataSource name="Q_1" defaultGroupName="G_DEPARTMENTS">
      <select>
```

```

        select * from departments
      </select>
    </dataSource>
    <dataSource name="Q_2" defaultGroupName="G_EMPLOYEES">
      <select>
        select * from employees
      </select>
    </dataSource>
  </data>
</report>

```

10.3.2 データ・ソース間のリンク

複数のデータ・ソースが存在するときは、それらのデータ・ソースをリンクして適切なデータ・モデルを作成することが必要となる場合もあります。Reports XML では、レポート・データ・モデル・リンク・オブジェクトも公開されています。このオブジェクトは、グループレベルおよび列レベルのリンクをサポートします。任意の数のリンクを指定して、必要なデータ・モデルを作成できます。

link 要素のデータ型定義は、次のとおりです。

```

<!ELEMENT link EMPTY>
<!--ATTLIST link
  name CDATA #IMPLIED
  parentGroup CDATA #IMPLIED
  parentColumn CDATA #IMPLIED
  childQuery CDATA #IMPLIED
  childColumn CDATA #IMPLIED
  condition (eq|lt|neq|gt|gteq|like|notLike) "eq"
  sqlClause (startWith|having|where) "where">

```

link 要素は、*data* 要素の内部に配置され、*data* 要素内に定義されている *dataSource* オブジェクトのうち任意の 2 つをリンクします。次に例を示します。

```

<report name="anyname" DTDVersion="9.0.2.0.0">
  <data>
    <dataSource name="Q_1" defaultGroupName="G_DEPARTMENTS">
      <select>
        select * from departments
      </select>
    </dataSource>
    <dataSource name="Q_2" defaultGroupName="G_EMPLOYEES">
      <select>
        select * from employees
      </select>
    </dataSource>
    <link name="L_1" parentGroup="G_DEPARTMENTS"
      parentColumn="DEPARTMENT_ID" childQuery="Q_2" childColumn="DEPARTMENT_

```

```

        ID1" condition="eq" sqlClause="where"/>
    </data>
</report>

```

link 要素の内部では、Reports のデフォルト設定メカニズムによって、DEPARTMENT_ID1 が EMPLOYEES 表における DEPARTMENT_ID 列の別名と認識されます。この場合、ユーザーが明示的に別名を作成する必要はありません。

10.3.3 各データ・ソース内でのグループ階層の作成

Oracle9iAS Reports Services では、完全なグループ階層を利用することができます。各グループ内ですべての列を指定し、それらの列の順序をブレイクすることができます。式、サマリーおよびプレースホルダを使用して、グループ内でさらにオブジェクトをカスタマイズできます。

group 要素のデータ型定義は、次のとおりです。

```

<!ELEMENT group
  (field|exception|rowDelimiter|xmlSettings|displayInfo|dataItem|formula|
  summary|placeholder|filter|comment)*>
<!ATTLIST group
  name CDATA #IMPLIED
  fillColor CDATA #IMPLIED
  lineColor CDATA #IMPLIED
  formatTrigger CDATA #IMPLIED>

```

次の例では、*group* 要素を使用して、データ・ソースの下にブレイク・グループを作成しています。

```

<report name="anyname" DTDVersion="9.0.2.0.0">
  <data>
    <dataSource name="Q_1">
      <select>
        select * from employees
      </select>
      <group name="G_DEPARTMENTS">
        <dataItem name="DEPARTMENT_ID"/>
      </group>
      <group name="G_EMPLOYEES">
        <dataItem="EMPLOYEE_ID"/>
        <dataItem="FIRST_NAME"/>
        <dataItem="LAST_NAME"/>
        <dataItem="JOB_ID"/>
        <dataItem="MANAGER_ID"/>
        <dataItem="HIRE_DATE"/>
        <dataItem="SALARY"/>
        <dataItem="COMMISSION_PCT"/>
      </group>
    </dataSource>
  </data>
</report>

```

```
        </group>
      </dataSource>
    </data>
  </report>
```

10.3.4 クロス積（マトリックス）グループの作成

クロス積グループを使用すると、データ・モデルに任意の数のグループのマトリックスを定義できます。クロス積の各ディメンション・グループは、同じデータ・ソースに属している場合でも、様々なデータ・ソースから結合されている場合でも、マトリックスを作成できます。この柔軟性に対応して、<crossProduct> タグは、すべてのデータ・ソースとグループが作成された後で、<data> タグ内に配置されます。

crossProduct 要素のデータ型定義は、次のとおりです。

```
<!ELEMENT crossProduct
  (xmlSettings|displayInfo|dimension|(formula|summary|placeholder)*|comment)*>
<ATTLIST crossProduct
  name CDDATA #IMPLIED
  mailText CDDATA #IMPLIED>
```

次の例では、1 回の問合せのマトリックスを作成しています。

```
<report name="anyname" DTDVersion="9.0.2.0.0">
  <data>
    <dataSource name="Q_1">
      <select>
        select * from employees
      </select>
      <group name="G_DEPARTMENTS">
        <dataItem name="DEPARTMENT_ID"/>
      </group>
      <group name="G_JOB_ID">
        <dataItem name="JOB_ID"/>
      </group>
      <group name="G_MANAGER_ID">
        <dataItem name="MANAGER_ID"/>
      </group>
      <group name="G_EMPLOYEE_ID">
        <dataItem name="EMPLOYEE_ID"/>
        <dataItem name="FIRST_NAME"/>
        <dataItem name="LAST_NAME"/>
        <dataItem name="HIRE_DATE"/>
        <dataItem name="SALARY"/>
        <dataItem name="COMMISSION_PCT"/>
      </group>
    </dataSource>
    <crossProduct name="G_Matrix">
```



```

        <dimension>
            <group name="G_DEPARTMENTS">
        </dimension>
        <dimension>
            <group name="G_JOB_ID">
        </dimension>
        <dimension>
            <group name="G_MANAGER_ID">
        </dimension>
    </crossProduct>
</data>
</report>

```

10.3.5 あらゆるレベルでの式、サマリーおよびプレースホルダの作成

式、サマリーおよびプレースホルダは、データ・モデル内のあらゆるレベルに配置できます。さらに、これらの各オブジェクトの属性は、どれも完全に制御することができます。

次の例では、ソースがグループ・レベルの式列に基づいている、レポートレベルのサマリーを作成しています。

```

<report name="anyname" DTDVersion="9.0.2.0.0">
    <data>
        <dataSource name="Q_1">
            <select>
                select * from employees
            </select>
            <group name="G_EMPLOYEES">
                <dataItem name="EMPLOYEE_ID"/>
                <dataItem name="EMPLOYEE_ID"/>
                <dataItem name="FIRST_NAME"/>
                <dataItem name="LAST_NAME"/>
                <dataItem name="HIRE_DATE"/>
                <dataItem name="SALARY"/>
                <dataItem name="COMMISSION_PCT"/>
                <dataItem name="DEPARTMENT_ID"/>
                <formula name="CF_REMUNERATION" source="cf_1formula"
                    datatype="number" width="20" precision="10"/>
            </group>
        </dataSource>
        <summary name="CS_REPORT_LEVEL_SUMMARY" function="sum" width="20"
            precision="10" reset="report" compute="report"/>
    </data>
    <programUnits>
        <function name="cf_1formula" returnType="number">
            <textSource>
                <![CDATA[

```

```

        function CF_1Formula return Number is
        begin
            return (:salary + nvl(:commission_pct,0));
        end;
    ]]>
</textSource>
</function>
</programUnits>
</report>

```

10.3.6 パラメータの作成

Reports XML では、*parameter* 要素が *data* 要素の開始と終了の間に配置されます。*parameter* 要素のデータ型定義は、次のとおりです。

```

<!ELEMENT parameter (comment?|listOfValues?)>
<!--ATTLIST parameter
    name CDATA #REQUIRED
    datatype (number|character|date) "number"
    width CDATA "20"
    scale CDATA "0"
    precision CDATA "0"
    initialValue CDATA #IMPLIED
    inputMask CDATA #IMPLIED
    validationTrigger CDATA #IMPLIED
    label CDATA #IMPLIED
    defaultWidth CDATA #IMPLIED
    defaultHeight CDATA #IMPLIED-->

```

次の例は、動的値リスト（LOV）、初期値および有効化トリガーを示しています。

```

<report name="anyname" DTDVersion="9.0.2.0.0">
  <data>
    <dataSource name="Q_1" defaultGroupName="G_DEPARTMENTS">
      <select>
        select * from departments
      </select>
    </dataSource>
    <parameter name="P_LAST_NAME" datatype="character" precision="10"
      initialValue="SMITH" validationTrigger="p_last_namevalidtrigger"
      defaultWidth="0" defaultHeight="0">
      <listOfValues restrictToList="yes">
        <selectStatement hideFirstColumn="yes">
          <![CDATA[select last_name, 'last_name||'-'||employee_id'
            from employees]]>
        </selectStatement>
      </listOfValues>
    </parameter>
  </data>
</report>

```

```

        </parameter>
    </data>
    <programUnits>
        <function name="p_last_namevalidtrigger" returnType="character">
            <textSource>
                <![CDATA[function P_LAST_NAMEValidTrigger return boolean is last_name char(20);
begin
    select count(*) into last_name from employees
    where upper(last_name)=upper(:p_last_name);
    exception when OTHERS then return(FALSE);
end;
return(TRUE);
end;
]]>
            </textSource>
        </function>
    </programUnits>
</report>

```

10.4 実行時の XML ファイルの使用方法

作成した Reports XML カスタマイズ・ファイルは、次の方法で使用できます。

- CUSTOMIZE コマンドライン引数または SRW.APPLY_DEFINITION ビルトインを指定して、実行時に XML レポート定義を RDF またはその他の XML ファイルに適用します。詳細は、「[実行時の XML レポート定義の適用](#)」を参照してください。
- REPORT（または MODULE）コマンドライン引数を指定して、XML レポート定義を単独で（他のレポートなしで）実行します。詳細は、「[XML レポート定義の単独での実行](#)」を参照してください。
- CUSTOMIZE コマンドライン引数を使用して、RWCONVERTER によるバッチ変更を実行します。詳細は、「[バッチ変更の実行](#)」を参照してください。

この後の項では、各ケースを詳しく説明するとともに、例を示します。

10.4.1 実行時の XML レポート定義の適用

実行時に XML レポート定義を RDF または XML ファイルに適用するには、CUSTOMIZE コマンドライン引数または SRW.APPLY_DEFINITION ビルトインを使用します。CUSTOMIZE は、RWCLIENT、RWRUN、RWBUILDER、RWCONVERTER および URL レポート要求とともに使用できます。

注意： CUSTOMIZE を RWCONVERTER とともに使用する方法の詳細は、「[バッチ変更の実行](#)」を参照してください。

10.4.1.1 1 つの XML レポート定義の適用

次に示すコマンドラインは、ジョブ要求を Oracle9iAS Reports Services に送信し、XML レポート定義 (EMP.XML) を RDF ファイル (EMP.RDF) に適用します。この例では、CUSTOMIZE コマンドが Windows のディレクトリ・パスにあるファイルを参照しています。UNIX の場合は、UNIX の標準に従ってパスを指定してください (つまり myreports/emp.xml)。

```
RWCLIENT REPORT=emp.rdf CUSTOMIZE=%myreports%emp.xml
  USERID=<username>/<password>@<my_db> DESTYPE=file DESNAME=emp.pdf
  DESFORMAT=PDF SERVER=<server_name>
```

Oracle9iAS Reports Services のランタイム・コマンドである RWRUN を使用する場合は、コマンドラインは、次のとおりです。

```
RWRUN USERID=<username>/<password>@<my_db> REPORT=emp.rdf
  CUSTOMIZE=%myreports%emp.xml DESTYPE=file DESNAME=emp.pdf
  DESFORMAT=PDF
```

XML レポート定義をテストするときは、トレース・ファイルを作成する引数を追加してレポート要求を実行すると便利な場合があります。次に例を示します。

```
TRACEFILE=emp.log TRACEMODE=trace_replace TRACEOPT=trace_app
```

注意： Reports Server 構成ファイルにデフォルトのトレース・オプションを指定した場合は、そのデフォルトを変更する必要がない限り、コマンドラインにトレースを含める必要はありません。

トレース・ファイルにより、レポート・オブジェクトの作成および書式に関する詳細なリストが得られます。

10.4.1.2 複数の XML レポート定義の適用

実行時に複数の XML レポート定義をレポートに適用するには、CUSTOMIZE コマンドライン引数にリストを指定します。次に示すコマンドラインは、2 つの XML レポート定義 EMP0.XML および EMP1.XML を RDF ファイル EMP.RDF に適用するジョブ要求を、Oracle9iAS Reports Services に送信します。

```
RWCLIENT REPORT=emp.rdf
  CUSTOMIZE="(D:%CORP%MYREPORTS%EMP0.XML,D:%CORP%MYREPORTS%EMP1.XML)"
  USERID=<username>/<password>@<my_db> DESTYPE=file DESNAME=emp.pdf
  DESFORMAT=PDF SERVER=<server_name>
```

注意： この例では、CUSTOMIZE コマンド・エントリに、Windows プラットフォームに格納されているファイルへのディレクトリ・パスが示されています。UNIX の場合は、そのプラットフォームの標準に従ってディレクトリ・パスを指定してください（つまり円記号（¥）のかわりに通常のスラッシュを使用）。

Oracle9iAS Reports Services Runtime を使用する場合は、次のとおりです。

```
RWRUN REPORT=emp.rdf
      CUSTOMIZE=" (D:¥CORP¥MYREPORTS¥EMP0.XML,D:¥CORP¥MYREPORTS¥EMP1.XML) "
      USERID=<username>/<password>@<my_db> DESTYPE=file DESNAME=emp.pdf
      DESFORMAT=PDF
```

10.4.1.3 PL/SQL での XML レポート定義の適用

PL/SQL で RDF ファイルに XML レポート定義を適用するには、BeforeForm または AfterForm トリガーで SRW.APPLY_DEFINITION および SRW.ADD_DEFINITION ビルトインを使用します。この後の項では、これらのビルトインの例を示します。

10.4.1.3.1 ファイルに保存された XML 定義の適用 ファイル・システムに保存された XML をレポートに適用するには、レポートの BeforeForm または AfterForm トリガーで SRW.APPLY_DEFINITION ビルトインを使用します。

Windows の場合：

```
SRW.APPLY_DEFINITION ('%<ORACLE_HOME>%¥TOOLS¥DOC¥US¥RBBR¥COND.XML');
```

UNIX の場合：

```
SRW.APPLY_DEFINITION ('$<ORACLE_HOME>/TOOLS/DOC/US/RBBR/COND.XML');
```

レポートの実行時に、トリガーが実行され、指定した XML ファイルがレポートに適用されます。

10.4.1.3.2 メモリーに保存された XML 定義の適用 メモリー内に XML レポート定義を作成するには、SRW.ADD_DEFINITION を使用してドキュメント・バッファに定義を追加した後に、SRW.APPLY_DEFINITION を使用してその定義を適用する必要があります。

次の例では、ユーザーが入力したパラメータ値に基づいてメモリー内にいくつかの定義を作成し、それらを適用する方法を示します。この例の PL/SQL は、videosales_custom.rdf というレポートの AfterParameterForm トリガーで使用されます。

videosales_custom.rdf ファイルの AfterParameterForm トリガーに入っている PL/SQL が実行する内容は、次のとおりです。

- 実行時にユーザーによって入力されたパラメータ値に基づいて、条件と合致したフィールドをハイライトします。
- 実行時にユーザーによって入力されたパラメータ値に基づいて、数値書式マスクを変更します。

次のヒントは、この例を見る場合に役立ちます。

- SRW.APPLY_DEFINITION を使用するたびにドキュメント・バッファがフラッシュされるので、そのたびに SRW.ADD_DEFINITION を使用して XML レポート定義を新規作成する必要があります。
- パラメータ hilite_profits、hilite_costs、hilite_sales および money_format を使用して、XML レポート定義に入れる内容を決定している点に注意してください。hilite_profits、hilite_costs および hilite_sales パラメータは、書式例外でも使用されており、これによってハイライトする値が決定します。
- VARCHAR2 列のサイズには上限があるため (4000 バイト)、かなり大きな XML レポート定義の場合は、複数の列にまたがって指定しなければならない場合があります。その場合は、1 つの大きな定義を作成して一度に適用するのではなく、メモリー内に定義をいくつか作成し、別々に適用する必要があります。

```
function AfterPForm return boolean is
begin
SRW.ADD_DEFINITION('<report name="vidsales_masks"
author="Generated" DTDVersion="9.0.2.0.0">');
IF :MONEY_FORMAT=' $NNNN.00' THEN SRW.ADD_DEFINITION('<layout>');
    SRW.ADD_DEFINITION('<section name="main">');
    SRW.ADD_DEFINITION('<field name="F_TOTAL_PROFIT" source="TOTAL_PROFIT"
formatMask="LNNNNNNNNNNN0D00"/>');
    SRW.ADD_DEFINITION('<field name="F_TOTAL_SALES" source="TOTAL_SALES"
formatMask="LNNNNNNNNNNN0D00"/>');
    SRW.ADD_DEFINITION('<field name="F_TOTAL_COST" source="TOTAL_COST"
formatMask="LNNNNNNNNNNN0D00"/>');
    SRW.ADD_DEFINITION('<field name="F_SumTOTAL_PROFITPerCITY" source="SumTOTAL_
PROFITPerCITY"
formatMask="LNNNNNNNNNNN0D00"/>');
    SRW.ADD_DEFINITION('<field name="F_SumTOTAL_SALESPerCITY" source="SumTOTAL_
SALESPerCITY"
formatMask="LNNNNNNNNNNN0D00"/>');
    SRW.ADD_DEFINITION('<field name="F_SumTOTAL_COSTPerCITY" source="SumTOTAL_
COSTPerCITY"
formatMask="LNNNNNNNNNNN0D00"/>');
    SRW.ADD_DEFINITION('</section>');
    SRW.ADD_DEFINITION('</layout>');
ELSIF :MONEY_FORMAT=' $NNNN' THEN SRW.ADD_DEFINITION('<layout>');
```

```

SRW.ADD_DEFINITION('<section name="main">');
SRW.ADD_DEFINITION('<field name="F_TOTAL_PROFIT" source="TOTAL_PROFIT"
formatMask="LNNNNNNNNNN0"/>');
SRW.ADD_DEFINITION('<field name="F_TOTAL_SALES" source="TOTAL_SALES"
formatMask="LNNNNNNNNNN0"/>');
SRW.ADD_DEFINITION('<field name="F_TOTAL_COST" source="TOTAL_COST"
formatMask="LNNNNNNNNNN0"/>');
SRW.ADD_DEFINITION('<field name="F_SumTOTAL_PROFITPerCITY" source="SumTOTAL_
PROFITPerCITY" formatMask="LNNNNNNNNNN0"/>');
SRW.ADD_DEFINITION('<field name="F_SumTOTAL_SALESPerCITY" source="SumTOTAL_
SALESPerCITY" formatMask="LNNNNNNNNNN0"/>');
SRW.ADD_DEFINITION('<field name="F_SumTOTAL_COSTPerCITY" source="SumTOTAL_
COSTPerCITY" formatMask="LNNNNNNNNNN0"/>');
SRW.ADD_DEFINITION('</section>');
SRW.ADD_DEFINITION('</layout>');
END IF;
SRW.ADD_DEFINITION('</report>');
SRW.APPLY_DEFINITION;
SRW.ADD_DEFINITION('<report name="vidsales_hilite_costs" author="Generated"
DTDVersion="9.0.2.0.0">');
IF :HILITE_COSTS <> 'None' THEN SRW.ADD_DEFINITION('<layout>');
    SRW.ADD_DEFINITION('<section name="main">');
    SRW.ADD_DEFINITION('<field name="F_TOTAL_COST" source="TOTAL_COST">');
    SRW.ADD_DEFINITION('<exception textColor="red">');
    SRW.ADD_DEFINITION('<condition source="TOTAL_COST" operator="gt"
operand1=":hilite_costs"/>');
    SRW.ADD_DEFINITION('</exception>');
    SRW.ADD_DEFINITION('</field>');
    SRW.ADD_DEFINITION('</section>');
    SRW.ADD_DEFINITION('</layout>');
END IF;
SRW.ADD_DEFINITION('</report>');
SRW.APPLY_DEFINITION;
SRW.ADD_DEFINITION('<report name="vidsales_hilite_sales" author="Generated"
DTDVersion="9.0.2.0.0">');
IF :HILITE_SALES <> 'None' THEN SRW.ADD_DEFINITION('<layout>');
    SRW.ADD_DEFINITION('<section name="main">');
    SRW.ADD_DEFINITION('<field name="F_TOTAL_SALES" source="TOTAL_SALES">');
    SRW.ADD_DEFINITION('<exception textColor="red">');
    SRW.ADD_DEFINITION('<condition source="TOTAL_SALES" operator="gt"
operand1=":hilite_sales"/>');
    SRW.ADD_DEFINITION('</exception>');
    SRW.ADD_DEFINITION('</field>');
    SRW.ADD_DEFINITION('</section>');
    SRW.ADD_DEFINITION('</layout>');
END IF;
SRW.ADD_DEFINITION('</report>');

```

```
SRW.APPLY_DEFINITION;
SRW.ADD_DEFINITION('<report name="vidsales_hilite_profits" author="Generated"
DTDVersion="9.0.2.0.0">');
IF :HILITE_PROFITS <> 'None' THEN SRW.ADD_DEFINITION('<layout>');
    SRW.ADD_DEFINITION('<section name="main">');
    SRW.ADD_DEFINITION('<field name="F_TOTAL_PROFIT" source="TOTAL_PROFIT">');
    SRW.ADD_DEFINITION('<exception textColor="red">');
    SRW.ADD_DEFINITION('<condition source="TOTAL_PROFIT" operator="gt"
operand1=":hilite_profits"/>');
    SRW.ADD_DEFINITION('</exception>');
    SRW.ADD_DEFINITION('</field>');
    SRW.ADD_DEFINITION('</section>');
    SRW.ADD_DEFINITION('</layout>');
END IF;
SRW.ADD_DEFINITION('</report>');
SRW.APPLY_DEFINITION;
return (TRUE);
end;
```

10.4.2 XML レポート定義の単独での実行

XML レポート定義を単独で実行するには、REPORT（または MODULE）引数に XML ファイルを指定して要求を送信します。次のコマンドラインは、レポート emp.xml を単独で実行するジョブ要求を Oracle9iAS Reports Services に送信します。

```
RWCLIENT USERID=<username>/<password>@<my_db>
REPORT=C:¥CORP¥MYREPORTS¥EMP.XML
DESTTYPE=file DESNAME=emp.pdf DESFORMAT=PDF
SERVER=<server_name>
```

Oracle9iAS Reports Services のランタイム・コマンドである RWRUN を使用する場合は、コマンドラインは、次のとおりです。

```
RWRUN USERID=<username>/<password>@<my_db>
REPORT=C:¥CORP¥MYREPORTS¥EMP.XML
DESTTYPE=file DESNAME=emp.pdf DESFORMAT=PDF
```

この方法で XML レポート定義を実行する場合は、必ず XML ファイルの拡張子を指定します。また、CUSTOMIZE 引数を使用して、XML カスタマイズ・ファイルをこのレポートに適用することもできます。

10.4.3 バッチ変更の実行

更新を必要とするレポートが大量にある場合は、RWCONVERTER で CUSTOMIZE コマンドライン引数を使用して、バッチ変更を実行できます。バッチ変更は、多数のレポートに同じ変更を繰り返し実行する必要がある場合に特に便利です（たとえば、フィールドの書式マスクの変更など）。Reports Builder でレポートを 1 つずつオープンして手動で変更するかわりに、RWCONVERTER を 1 回実行するだけで、一度に多数のレポートに同じ変更を加えることができます。

次の例では、2 つの XML レポート定義 translate.xml および customize.xml を 3 つの RDF ファイル INVEN1.RDF、INVEN2.RDF および MANU.RDF に適用し、変更後の定義を新しいファイル INVEN1_NEW.RDF、INVEN2_NEW.RDF および MANU_NEW.RDF に保存します。

```
RWCONVERTER <username>/<password>@<my_db>
  STYPE=rdffile
  SOURCE="(inven1.rdf, inven2.rdf, manu.rdf)"
  DTYPE=rdffile
  DEST="(inven1_new.rdf, inven2_new.rdf, manu_new.rdf)"
  CUSTOMIZE="(D:¥APPS¥TRANS¥TRANSLATE.XML,D:¥APPS¥CUSTOM¥CUSTOMIZE.XML)"
  BATCH=yes
```

注意： この例では、CUSTOMIZE コマンド・エントリに、Windows プラットフォームに格納されているファイルへのディレクトリ・パスが示されています。UNIX の場合は、そのプラットフォームの標準に従ってディレクトリ・パスを指定してください（つまり円記号（¥）のかわりに通常のスラッシュを使用）。

10.5 XML レポート定義のデバッグ

次の機能は、XML レポート・ファイルをデバッグする際に役立ちます。

- [XML パーサーのエラー・メッセージ](#)
- [トレース・オプション](#)
- [RWBUILDER](#)
- [デバッグを目的とした、ファイルへの XML の書込み](#)

次の各項では、これらの機能を説明します。

10.5.1 XML パーサーのエラー・メッセージ

XML パーサーは、コア Oracle データベース・リリースに付属する Oracle の XML Developer's Kit (XDK) に含まれています。XML パーサーは、XML 構文の有効性をチェックする Java パッケージです。XML パーサーを含む JAR ファイルは、インストール時に自動的に設定され、Reports で使用可能となります。

XML パーサーは、ほとんどの構文エラーを検出し、エラー・メッセージを表示します。エラー・メッセージは、エラーが発生した XML ファイルの行番号と、エラーの簡単な説明から成ります。

XML パーサーの詳細は、Oracle Technology Network (<http://otn.oracle.com/> 英語サイト) または Oracle Technology Network Japan (<http://otn.oracle.co.jp/> 日本語サイト) を参照してください。その場合、*XML parser* または *XDK* で検索します。Oracle データベース付属のドキュメントにも情報が記載されています。

10.5.2 トレース・オプション

XML レポート定義をテストする際に、トレース・ファイルを作成する引数を追加してレポートを実行すると便利です。次に例を示します。

```
RWRUN <username>/<password>@<my_db>
REPORT=¥CORP¥MYREPORTS¥EMP.XML
TRACEFILE=emp.log
TRACEMODE=trace_replace
TRACEOPT=trace_app
```

このコマンドラインの最後の 3 つの引数によって、レポート処理の詳細リストを含むトレース・ファイルが生成されます。トレース・ファイル・ログのデフォルトの場所は、Windows と UNIX で同じです。

```
ORACLE_HOME¥reports¥logs¥
```

注意： この例では、REPORT コマンド・エン트리およびトレース・ログへのパスに、Windows プラットフォームに格納されているファイルへのディレクトリ・パスが示されています。UNIX の場合は、そのプラットフォームの標準に従ってディレクトリ・パスを指定してください（つまり円記号 (¥) のかわりに通常のスラッシュを使用）。

10.5.3 RWBUILDER

XML レポート定義の設計時には、**Reports Builder** でオープンした方が便利な場合があります。Reports Builder では、オブジェクトが期待どおりに作成または変更されているかをすぐに確認できます。たとえば、XML レポート定義にサマリーを作成する場合、Reports Builder で定義をオープンすると、サマリーがデータ・モデルの適切なグループに配置されているかどうかが目で見えます。

Reports Builder で完全なレポート定義をオープンするには、REPORT（または MODULE）キーワードを使用します。次に例を示します。

```
RWBUILDER USERID=<username>/<password>@<my_db>
REPORT=C:¥CORP¥MYREPORTS¥EMP.XML
```

Oracle9iDS Reports Services Builder で部分レポート定義をオープンするには、CUSTOMIZE キーワードを使用します。次に例を示します。

```
RWBUILDER USERID=<username>/<password>@<my_db> REPORT=EMP.RDF
CUSTOMIZE=C:\MYREPORTS\EMP.XML
```

注意： この例では、REPORT コマンド・エントリに、Windows プラットフォームに格納されているファイルへのディレクトリ・パスが示されています。UNIX の場合は、そのプラットフォームの標準に従ってディレクトリ・パスを指定してください（つまり円記号（¥）のかわりに通常のスラッシュを使用）。

どちらの場合も、Reports Builder で有効な XML レポート定義がオープンします。その後、Oracle9iAS Reports Services Editor の様々なビューを使用して、レポートが期待どおりに作成または変更されているかどうかを確認できます。

10.5.4 デバッグを目的とした、ファイルへの XML の書込み

SRW.ADD_DEFINITION を使用して XML レポート定義をメモリー内に作成する場合は、XML をファイルに書込むとデバッグに役立ちます。次の例は、各行をメモリー内のドキュメント・バッファおよび（ファイルへの書込みを選択した場合は）指定したファイルに書き込むプロシージャを示しています。

```
PROCEDURE addaline (newline VARCHAR, outfile Text_IO.File_Type) IS
BEGIN
    SRW.ADD_DEFINITION(newline);
    IF :WRITE_TO_FILE='Yes' THEN
        Text_IO.Put_Line(outfile, newline);
    END IF;
END;
```

この例が実際に機能するには、プロシージャを呼び出す PL/SQL で TEXT_IO.File_Type 型の変数を宣言する必要があります。次に例を示します。

```
custom_summary Text_IO.File_Type;
```

また、書込み用のファイルをオープンし、addaline プロシージャを呼び出して、書き込む文字列と書込み先ファイルを渡す必要があります。次に例を示します。

```
custom_summary := Text_IO.Fopen(:file_directory || 'vid_summ_per.xml', 'w');
addaline('<report name="video_custom" author="Generated" DTDVersion="9.0.2.0.0">',
        custom_summary);
```

イベント・ドリブンによる公開

現代のビジネス・プロセスでは、作業環境に自動化を取り入れる必要性が頻繁に生じますが、自動化は、ファンクションやプロシージャを背後で起動することによって実現します。背後で実行されるタスクには、発注が処理される際に自動的に印刷される請求書などの出力の自動生成、最新データで自動的に更新される Web サイト、トランザクション完了時の最新レポートの自動メール送信などが挙げられます。

イベントに応じた自動出力は、かつてはかなり複雑な作業でした。特に、Oracle9i Reports Developer のような対話型の RAD 開発ツールを使用して、可能な限り同じ結果を得ようとする場合は特に困難でした。

自動出力の要求に対応するために、Oracle では Oracle9iAS Reports Services にスケジューリング・メカニズムを導入し、ユーザーの操作を追加することなく定期的にレポートを起動できるようにしました。ただし、これでも 1 つだけ解決されない要求事項が残りました。それは、レコードの挿入や値の変更といったデータベース内のイベントに応じて、自動的にレポートを実行する機能です。

Oracle9iAS Reports Services の Event-Driven Publishing API を使用すると、レコードの挿入や値の変更といったデータベース内のイベントに応じて自動的にレポートを実行できます。Event-Driven Publishing API は PL/SQL API で、データベースから Oracle9iAS Reports Services へのジョブの自動送信を可能にします。

この章では、Event-Driven Publishing API について説明するとともに、その使用例を示します。この章には、次の項があります。

- [Event-Driven Publishing API](#)
- [Event-Driven Publishing API を使用する アプリケーションのデバッグ](#)
- [データベース・イベントからのレポートの起動](#)
- [Oracle9i Advanced Queuing との統合](#)

11.1 Event-Driven Publishing API

Event-Driven Publishing API は、データベース内のイベントに応じるプロシージャの開発に必要な、基本的なファンクションを提供する PL/SQL パッケージです。イベント・ドリブン・ジョブは、HTTP プロトコルを使用して送信されます。サーバーは、すべてのコールに一意の `job_ident` レコードを割り当てます。これは、ジョブのステータスを追跡する際に役立ちます。

11.1.1 API の要素

この API は、いくつかの重要な要素から構成されています。

- **SRW-Package** には、ジョブの送信、ジョブ・ステータスのチェック、ジョブの取消しおよびパラメータ・リストの操作に関連するすべてのファンクションとプロシージャが含まれています。
- **ParamList-Type** は、パラメータ・リストを定義します。パラメータ・リストは、ジョブを送信する際に値を渡すための主要な手段です。パラメータ・リストは、ジョブ送信ごとに必要となります。このリストには、複数の重要なパラメータが含まれています。
- **ParamList-Object** は、アドバンスド・キューイングなどの機能に必要です。これらの機能では、パラメータ・リストをデータベースに格納して、メッセージとともに渡せるようにする必要があります。

API の各要素については、この後の項で詳しく説明します。

この API は、Oracle9iAS Reports Services Security および Oracle9iAS Portal とともにインストールされますが、どちらも必須ではありません。Oracle9iAS Portal を使用しないデータベースにも API をインストールできるように、インストール・スクリプトが別途用意されています。

- **srwAPIins.sql** は、Event-Driven Publishing API をインストールします。
- **srwAPIgrant.sql** は、API へのアクセス権限を付与します。このスクリプトは、API へのアクセス権限を付与するユーザーごとに実行します。全員がアクセス可能である場合は、このスクリプトを 1 回実行し、PUBLIC にアクセス権限を付与できます。
- **srwAPIdrop.sql** は、API を削除します。

11.1.2 パラメータ・リストの作成と操作

パラメータ・リストは、SRW_PARAMLIST タイプの PL/SQL 変数です。このタイプの変数は、SRW_PARAMETER タイプの 255 個の要素から成る配列です。SRW_PARAMETER タイプの要素自体は、2 つの属性 NAME および VALUE から構成されます。API には、パラメータ・リストを操作するための次のプロシージャが用意されています。

- [Add_Parameter](#)
- [Remove_Parameter](#)
- [Clear_Parameter_List](#)

これらのプロシージャを使用して、パラメータ・リストを操作できます。この項では、これらのプロシージャを簡単に説明します。詳細は、Oracle9iAS Reports API のドキュメントを参照してください。

注意： Reports API の最新情報は、Oracle Technology Network (<http://otn.oracle.com/> 英語サイト) または Oracle Technology Network Japan (<http://otn.oracle.co.jp/> 日本語サイト) を参照してください。

11.1.2.1 Add_Parameter

パラメータ・リストを初めて使用するときは、そのパラメータ・リストを初期化してからパラメータを追加する必要があります。次に例を示します。

```
DECLARE
    myPlist SRW_PARAMLIST;
BEGIN
    myPlist := SRW_PARAMLIST(SRW_PARAMETER('',''));
    srw.add_parameter(myPlist,'myParameter','myValue');
END;
```

パラメータの属性 (NAME と VALUE) はどちらも VARCHAR2 タイプで、その長さは NAME が 80 文字、VALUE が 255 文字に制限されています。

ADD_PARAMETER ファンクションには、オプションで MODE という 3 番目の属性を指定できます。MODE は、同じ名前のパラメータがすでに存在する場合に、そのパラメータを上書きするか、またはエラーが発生するかを決定します。重複する名前があった場合にエラーが発生するように指定するには、定数 CHECK_FOR_EXISTANCE を使用します。これは、MODE 属性のデフォルト値です。重複する名前があった場合にパラメータを上書きするように指定するには、定数 OVERWRITE_IF_EXISTS を使用します。

11.1.2.2 Remove_Parameter

REMOVE_PARAMETER は、パラメータ・リストからパラメータを削除するために使用します。このプロシージャを呼び出し、対象となるパラメータ・リストとともに削除するパラメータの名前を渡します。

次に例を示します。

```
DECLARE
    myPlist SRW_PARAMLIST;
BEGIN
    myPlist := SRW_PARAMLIST(SRW_PARAMETER('',''));
    srw.add_parameter(myPlist,'myParameter','myValue');
    srw.remove_parameter(myPlist,'myParameter');
END;
```

11.1.2.3 Clear_Parameter_List

リストからすべてのパラメータを削除するには、CLEAR_PARAMETER_LIST を使用します。次に例を示します。

```
DECLARE
    myPlist SRW_PARAMLIST;
BEGIN
    myPlist := SRW_PARAMLIST(SRW_PARAMETER('',''));
    srw.add_parameter(myPlist,'myParameter','myValue');
    srw.clear_parameter_list(myPlist);
END;
```

これによって、リストからすべてのパラメータが削除されます。

11.1.3 ジョブの送信方法

パラメータ・リストには、ジョブの送信に不可欠なすべてのパラメータが含まれています。ジョブ・タイプによって、リストに必要なパラメータが決まり、Reports Server で要求を処理できるようになります。

リスト内のパラメータは、ブラウザから Reports Servlet にジョブを送信する際に指定するパラメータと同じです。この場合、ジョブがレポートであるときは、少なくとも次のパラメータが必要です（それ以上指定してもかまいません）。

- GATEWAY は、要求の処理に使用する Reports Servlet への URL を提供します。
- SERVER は、サーブレットとともに使用される Reports Server を特定します。
- REPORT は、実行するレポート・ファイルを特定します。
- USERID は、レポートを実行するユーザーの名前と ID を特定します。

- AUTHID は、セキュリティで保護されたサーバーにアクセスする場合の認証情報を提供します。

各要求からは、`job_ident` レコードが返されます。このレコードには、ジョブを一意に識別するための情報が含まれています。この情報は、`SRW.JOB_IDENT` タイプの変数に格納されます。この情報は `PACKAGE-TYPE` であり、`SRW.JOB_IDENT` として参照する必要があります。ことに注意してください。一方、パラメータ・リストは `OBJECT-TYPE` であり、`SRW_PARAMLIST` として参照する必要があります。

次に例を示します。

```
DECLARE
    myPlist SRW_PARAMLIST;
    myIdent SRW.Job_Ident;
BEGIN
    myPlist := SRW_PARAMLIST(SRW_PARAMETER('',''));
    srw.add_parameter(myPlist,'GATEWAY','http://...');
    srw.add_parameter(myPlist,'SERVER','mySVR');
    srw.add_parameter(myPlist,'REPORT','myReport.RDF');
    srw.add_parameter(myPlist,'USERID','me/secret');
    myIdent := srw.run_report(myPlist);
END;
```

API の `RUN_REPORT` メソッドは、必要不可欠なすべての情報を含んだパラメータ・リストを入力として (`ADD_PARAMETER` を使用)、要求を作成して送信するとともに、`job_ident` レコードを返します。

`job_ident` レコードには、次のパラメータが含まれます。

- `MyIdent.GatewayURL`
- `MyIdent.ServerName`
- `MyIdent.JobID`
- `MyIdent.AuthID`

これらのパラメータは、`SRW.REPORT_STATUS` ファンクションが、送信されたジョブのステータス情報を取得するために必要となります。

11.1.4 ステータスの確認方法

Event-Driven Publishing API は、Reports Server との双方向通信を実現します。ユーザーは、サーバーにジョブを送信するとともに、SRW.REPORT_STATUS ファンクションを使用して、このジョブのステータスをサーバーに問い合わせることができます。

このファンクションからは、SRW.STATUS_RECORD タイプのレコードが返されます。このレコードには、Reports Servlet の SHOWJOBS コマンドを使用したときにジョブ・ステータス表示に示される情報と同じ情報が含まれています。

次に例を示します。

```
DECLARE
    myPlist SRW_PARAMLIST;
    myIdent SRW.Job_Ident;
    myStatus SRW.Status_Record;
BEGIN
    myPlist := SRW_PARAMLIST(SRW_PARAMETER('',''));
    srw.add_parameter(myPlist, 'GATEWAY', 'http://...');
    srw.add_parameter(myPlist, 'SERVER', 'mySVR');
    srw.add_parameter(myPlist, 'REPORT', 'MyReport.RDF');
    srw.add_parameter(myPlist, 'USERID', 'me/secret');
    myIdent := srw.run_report(myPlist);
    myStatus := srw.report_status(myIdent);
END;
```

返されたステータス・レコードを使用して、ジョブのステータスに関する情報をフェッチできます。

11.1.5 サーバーのステータス・レコードの使用方法

ステータス・レコードには、ジョブに関する処理情報が含まれています。これには、サーバー・キュー内の情報（SHOWJOBS で表示）と同じ情報が含まれます。さらに、終了したジョブに対して生成されたファイルに関する情報、およびスケジュールされたジョブの系統に関する情報も含まれます。

ステータス・レコードで最も重要な情報は、ランタイム・エラーとその原因をチェックするために使用される、現行ジョブ・ステータスとステータス・テキストです。

タイミング情報を使用して、完了までの予測時間を超過した場合にジョブが取り消されるかどうかを確認できます。

ステータス・レコードの用途の 1 つに、ジョブの取消しがあります。Event-Driven Publishing API では、サーバーに送信されたジョブを取り消す方法を提供しています。この方法は、許可されている実行時間を超過したジョブを削除する場合、または単にスケジュールされているジョブを取り消す場合に便利です。

ジョブを取り消すには、次のプロシージャを使用します。

```
DECLARE
    myPlist SRW_PARAMLIST;
    myIdent SRW.Job_Ident;
    myStatus SRW.Status_Record;
BEGIN
    myPlist := SRW_PARAMLIST(SRW_PARAMETER('',''));
    srw.add_parameter(myPlist,'GATEWAY','http://...');
    srw.add_parameter(myPlist,'SERVER','mySVR');
    srw.add_parameter(myPlist,'REPORT','myReport.RDF');
    srw.add_parameter(myPlist,'USERID','me/secret');
    myIdent := srw.run_report(myPlist);
    srw.cancel_report(myIdent);
END;
```

この例からわかるように、レポートを取り消す場合は、CANCEL_REPORT プロシージャ (srw.cancel_report) を呼び出し、取り消すジョブの job_ident レコードをそのプロシージャに渡します。このプロシージャでは、オプションのパラメータ・リストによって、必要な追加のパラメータを渡すことができます。

11.2 Event-Driven Publishing API を使用する アプリケーションのデバッグ

これまでに説明した処理はすべて背後で実行されるため、通常の実行ではデバッグ情報を生成できません。このため API には、特別なデバッグ・モードに切り替えるための 2 種類のプロシージャが用意されており、DBMS_OUTPUT によって拡張デバッグ情報が生成されます。

- SRW.START_DEBUGGING
- SRW.STOP_DEBUGGING

デバッグ・モードに切り替えるには、SRW.START_DEBUGGING を呼び出します。デバッグ・モードをオフにするには、SRW.STOP_DEBUGGING を呼び出します。デバッグ・モードは、実際のロジックを実行する直前に開始する必要があります。このモードは、パッケージの現行インスタンスがロードされている限り持続します。

この情報を表示する 1 つの方法として、スクリプトを実行する前に、SQL*PLUS で SERVEROUT を ON に設定します。

このデバッグ方法に加えて、API にはエラー処理に使用する一連の事前定義の例外が用意されています。Oracle9iAS Reports Services のインストールに含まれている `srw_test.sql` スクリプトには、これらの例外の例が示されています。これらの例外の詳細は、Reports API のリファレンス・ドキュメントを参照してください。

注意： Reports API の最新情報は、Oracle Technology Network (<http://otn.oracle.com/> 英語サイト) または Oracle Technology Network Japan (<http://otn.oracle.co.jp/> 日本語サイト) を参照してください。

11.3 データベース・イベントからのレポートの起動

データベース・トリガーは、Event-Driven Publishing API でレポートを起動するための主要なメカニズムです。Oracle データベースでは、各種イベントに応じて起動される様々な範囲のトリガーを定義できます。データベース・ドリブン・ジョブを送信するには、これまでの項で説明したコードを、データベース・トリガー内で使用します。

イベント・ドリブンの公開には、多数の用途があります。その1つとして、権限が付与されたとき、またはユーザーがログオン / ログオフしたときに起動されるトリガーを使用して、セキュリティ・プロトコルを作成できます。また、あるタイプの変更が表内のデータに加えられた場合に実行される自動処理を作成することもできます。たとえば、経費報告書のステータスが **DONE** (完了) に変わったときにデータベース・トリガーを起動し、それによって従業員の管理者にそのレポートを自動的に送信できます。

次に例を示します。

```
CREATE TRIGGER EXP_REP_TRG
  AFTER INSERT OR UPDATE on EXP_REP FOR EACH ROW
DECLARE
  myPlist SRW_PARAMLIST;
  myIdent SRW.Job_Ident;
BEGIN
  IF (:new.ExpStat = 'DONE') THEN
    myPlist := SRW_PARAMLIST(SRW_PARAMETER('',''));
    srw.add_parameter(myPlist,'GATEWAY','http://...');
    srw.add_parameter(myPlist,'SERVER','fooSVR');
    srw.add_parameter(myPlist,'REPORT','foo.RDF');
    srw.add_parameter(myPlist,'USERID','foo/bar');
    srw.add_parameter(myPlist,'ExpenseID',:new.ExpID);
    myIdent := srw.run_report(myPlist);
  END IF;
END;
```

このトリガーは、EXP_REP 表が更新されるたびに起動されます。ステータスが **DONE** に変わると、レポート要求が実行されます。

cgicmd.dat ファイルに指定されているキーを使用して要求を実行する場合は、REPORT パラメータのかわりに CMDKEY パラメータを指定します。このキーにユーザー ID 情報が含まれている場合は、USERID パラメータも省略できます。次に例を示します。

```
CREATE TRIGGER EXP_REP_TRG
  AFTER INSERT OR UPDATE on EXP_REP FOR EACH ROW
DECLARE
  myPlist SRW_PARAMLIST;
  myIdent SRW.Job_Ident;
BEGIN
  IF (:new.ExpStat = 'DONE') THEN
    myPlist := SRW_PARAMLIST(SRW_PARAMETER('',''));
    srw.add_parameter(myPlist,'GATEWAY','http://...');
    srw.add_parameter(myPlist,'SERVER','fooSVR');
    srw.add_parameter(myPlist,'CMDKEY','keyvalue');
    srw.add_parameter(myPlist,'ExpenseID',:new.ExpID);
    myIdent := srw.run_report(myPlist);
  END IF;
END;
```

さらに、配布 XML ファイルを使用した拡張配布モデルを定義した場合は、DIST パラメータにそのファイルを指定できます。次に例を示します。

```
CREATE TRIGGER EXP_REP_TRG
  AFTER INSERT OR UPDATE on EXP_REP FOR EACH ROW
DECLARE
  myPlist SRW_PARAMLIST;
  myIdent SRW.Job_Ident;
BEGIN
  IF (:new.ExpStat = 'DONE') THEN
    myPlist := SRW_PARAMLIST(SRW_PARAMETER('',''));
    srw.add_parameter(myPlist,'GATEWAY','http://...');
    srw.add_parameter(myPlist,'SERVER','fooSVR');
    srw.add_parameter(myPlist,'REPORT','foo.RDF');
    srw.add_parameter(myPlist,'USERID','foo/bar');
    srw.add_parameter(myPlist,'DISTRIBUTE','YES');
    srw.add_parameter(myPlist,'DESTINATION','filename.xml');
    srw.add_parameter(myPlist,'ExpenseID',:new.ExpID);
    myIdent := srw.run_report(myPlist);
  END IF;
END;
```

これは、この種のロジックをアプリケーションからデータベースに移し、データベースをビジネス・プロセスの中心的なストレージとして使用するための 1 つの方法です。

注意： Oracle9iAS Reports Services のインストールに含まれている `srw_test.sql` デモ・スクリプトには、Event-Driven Publishing API のその他の使用例が示されています。

11.4 Oracle9i Advanced Queuing との統合

Oracle Advanced Queuing は、キューと 2 つのプロセスを中心として非同期の要求 / 応答メカニズムを構築するための手段となります。2 つのプロセスとは、MESSAGES をキューに入れる ENQUEUE、およびキューを読み込む DEQUEUE です。

アドバンスト・キューイングは、キュー間でメッセージを配布し、キューをサブスクライブするための高度なメカニズムを備えています。これらのメカニズムは、いずれも基本要素 (ENQUEUE、DEQUEUE および MESSAGES) をベースに構築されています。

Event-Driven Publishing API では、これらのキューを使用してレポート・ジョブを保存および送信することができます。Oracle9iAS Reports Services で提供するメカニズムが要件に合わない場合は、独自のキューイング・メカニズムを作成することも可能です。

11.4.1 SRW_PARAMLIST タイプのメッセージを保持するキューの作成

キューは、いくつかの管理列とメッセージを表すオブジェクト列で構成されるデータベース内の表です。ここでのメッセージは、パラメータ・リストです。

アドバンスト・キューイング付属の `dbms_AQadm` パッケージには、アドバンスト・キューイング・システムの設定に必要なすべての管理ファンクションが含まれています。

データベースに物理的な表を作成するには、`dbms_AQadm.Create_Queue_Table` を使用します。その場合、表の名前、およびそのキューのメッセージを定義するオブジェクト・タイプの名前を渡す必要があります。

次に例を示します。

```
...
execute dbms_AQadm.Create_Queue_Table
(queue_Table=>'queuename_tab',
queue_Payload_Type=>'SRW_ParamList_Object',
compatible=>'9.0');
```

前述の例のオブジェクト表記で、SRW_PARAMLIST タイプをカプセル化するオブジェクト・タイプ SRW_PARAMLIST_OBJECT をすでに作成しているので、これをメッセージとして使用できます。

キュー・テーブルを作成したら、`dbms_AQadm.Create_Queue` でキューを作成し、`dbms_AQadm.Start_Queue` でキューを開始する必要があります。

次に例を示します。

```
...
execute dbms_AQadm.Create_Queue
    (Queue_Name=>'queueuname', Queue_Table=>'queueuname_tab');
prompt ... starting queue
execute dbms_AQadm.Start_Queue
    (Queue_Name=>'queueuname');
...
```

注意： Oracle9iAS Reports Services のインストレーションに含まれている `srwAQsetup.sql` デモ・ファイルには、単純なキューの設定、作成、開始の完全な例が示されています。

キューを作成して開始した後に必要となるのは、このキューにメッセージを作成するプロセスと、キューを読み込んでサーバーにジョブを送信するプロセスです。次の各項では、これらのプロセスを説明します。

11.4.2 エンキュー・プロセスの作成

エンキュー・プロセスは、キューにメッセージを挿入します。このプロセスは、データベース・トリガーによって呼び出されるアプリケーション、または外部のメカニズムを通じて提供されるアプリケーションに組み込むことができます。この項では、このキューに単純なメッセージを挿入するストアド・プロセスの作成例を示します。

ここでは、メッセージがパラメータ・リストそのものであるため、プロセスはかなり簡単です。パラメータ・リストを作成するには、前述の項で使ったコードを使用します。使用した変数に加えて、キューに入れるメッセージを保持する `object` 変数を定義します。

```
...
    plist_object SRW_ParamList_Object;
...
```

パラメータ・リストを作成したら、オブジェクト・コンストラクタで実際のメッセージ・オブジェクトを作成します。

```
...
    plist_object := SRW_ParamList_Object(plist);
...
```

その後に、アドバンスド・キューイング付属のエンキュー・プロセスでメッセージをエンキューします。

```
...
dbms_aq.enqueue(queue_name => 'myQueue',
  enqueue_options => enqueue_options,
  message_properties => message_properties,
  payload => PList_Object,
  msgid => message_handle);
...
```

メッセージがキューに挿入されます。メッセージ配布は一切設定していないため、次の項で説明するデキュー・プロシージャによってフェッチされるまで、メッセージはキューに入っただけとなります。

注意： `dbms_aq.enqueue` の正しい構文は、Advanced Queuing API のリファレンス・ドキュメントで確認してください。

Oracle9iAS Reports Services のインストールレーションに含まれている `srwAQsetup.sql` ファイルには、その他の例が示されています。

Reports API の最新情報は、Oracle Technology Network (<http://otn.oracle.com/> 英語サイト) または Oracle Technology Network Japan (<http://otn.oracle.co.jp/> 日本語サイト) を参照してください。

11.4.3 デキュー・プロシージャの作成

デキュー・プロシージャは、キュー内のすべてのメッセージを読み込んで処理します。ここでは、メッセージを読み込み、メッセージに添付されたパラメータ・リストを使用してサーバーにジョブを送信します。

この処理には、次の例を使用します。

```
BEGIN
  dequeue_options.wait := 1;
loop
  DBMS_AQ.DEQUEUE(queue_name => 'myQueue',
    dequeue_options => dequeue_options,
    message_properties => message_properties,
    payload => PList_Object,
    msgid => message_handle);
  COMMIT;
  plist := plist_object.params;
  r_jid := SRW.run_report(plist);
end loop;
exception when aq_timeout then
  begin
    NULL;
  end;
END;
```


このサンプル・コードでは、すべてのメッセージが処理されるまでキューが読み込まれます。処理に使用できる時間は、コードの 2 行目に定義されているタイムアウトによって決まります。このタイムアウトによって、タイムアウト例外が作成されるまでデキュー・プロシージャがメッセージを待機できる秒数が定義されます。

アドバンスド・キューイングには、メッセージを読み込むための `DBMS_AQ.DEQUEUE` ビルトインが用意されています。このビルトインは、情報を保持するオブジェクトであるメッセージのペイロードを、ペイロード・パラメータで定義されたオブジェクトに挿入します。

`plist` を使用して、ペイロード・オブジェクトから情報を抽出します。すでに説明したように、ここではオブジェクトにパラメータ・リストが含まれています。パラメータ・リストは、オブジェクト内部の `PARAMS` 属性に格納されています。抽出されたパラメータ・リストは、ジョブを送信する `SRW.RUN_REPORT` に渡されます。

このデキュー・プロシージャを手動で起動しないようにするには、データベース内部でジョブとして実行します。

第III部

各国語サポートおよび双方向サポート

第 III 部では、Reports 関連の各国語サポート設定および双方向サポートに関する情報を記載します。第 III 部には、次の章があります。

- [第 12 章「NLS と双方向サポート」](#)

NLS と双方向サポート

各国に配布するレポートを設計する場合は、キャラクタ・セットやテキストの読み込み順序などを考慮する必要があります。Oracle9iAS Reports Services では、これらの考慮事項に関連する問題に対処するためのサポートを提供しています。これらのサポートは、キャラクタ・セットに対する各国語サポート (NLS) と、テキストの読み込み順序に対する双方向サポートです。

Oracle NLS は、複数の異なる言語で展開できるアプリケーションの設計を可能にします。Oracle では、ヨーロッパ、中東およびアジアで使われている言語のほとんどをサポートしています。NLS を使用すると、次のことが可能になります。

- 各国のキャラクタ・セット (マルチバイト・キャラクタ・セットを含む) の使用
- 適切な言語と地域の表記規則に基づいたデータの表示
- インタフェースに表示される文字列の抽出とそれらの翻訳

双方向サポートは、利用対象者の要件に応じて、左から右または右から左の方向にデータを表示できるようにします。

この章では、NLS アーキテクチャを、Reports に関連する NLS の設定も含めて概説するとともに、JSP でキャラクタ・セットを指定する方法を説明します。また、Oracle9i でサポートされている双方向、Unicode および翻訳に関する情報も提供します。この章の主な項は、次のとおりです。

- [NLS アーキテクチャ](#)
- [NLS の環境変数](#)
- [JSP または XML ファイルでのキャラクタ・セットの指定](#)
- [双方向サポート](#)
- [Unicode](#)
- [アプリケーションの翻訳](#)

12.1 NLS アーキテクチャ

Oracle NLS アーキテクチャは、次の 2 つの部分で構成されます。

- 言語に依存しない機能
- 言語依存データ

12.1.1 言語に依存しない機能

言語に依存しない機能では、実行時にオペレータが使用する言語および地域に応じて、適切にデータが処理されます。データは、各国の日付および時刻の表記規則に従って、自動的に書式設定されます。

12.1.2 言語依存データ

言語依存データを使用すると、データを分離できます。これによって、作成したアプリケーションで、そのアプリケーション固有の文字列の翻訳を処理できるようになります。

言語依存データはコードから分離されているため、NLS 機能の動作は実行時に提供されるデータによって制御されます。新規言語を追加するとき、および言語別のアプリケーションの特性を変更するときに、コードを変更する必要はありません。このアーキテクチャでは、セッションごとに言語依存機能を指定することもできます。

12.2 NLS の環境変数

NLS の環境変数は、Oracle9iAS をインストールする際に、自動的にデフォルト値に設定されます。

注意： ある特定の Oracle9iAS Reports Services ホスト・マシンでは、言語を 1 つしか指定できません。使用する言語がそれぞれに異なる利用対象者にアプリケーション・サーバー・サービスを提供する場合は、言語ごとにホスト・マシンを用意する必要があります。

表 12-1 では、特に Oracle9iAS Reports Services に関する NLS 関連の環境変数を説明します。

注意： NLS のすべての環境変数の詳細は、『Oracle9i Database グローバリゼーション・サポート・ガイド』を参照してください。

表 12-1 Oracle9iAS Reports Services に特に関連する環境変数

変数	説明
NLS_LANG	Oracle9iAS Reports Services に関連し、Oracle9iAS Reports Services で使用される言語設定
DEVELOPER_NLS_LANG	Oracle9iDS Report Builder で使用される言語
USER_NLS_LANG	Oracle9iAS Reports Runtime コンポーネントで使用される言語

12.2.1 NLS_LANG 環境変数

NLS_LANG 環境変数は、Oracle9iAS Reports Services で使用される言語、地域およびキャラクタ・セットの設定を指定します。具体的な設定項目は次のとおりです。

- ユーザーに表示されるメッセージの言語
- DATE および NUMBER データ型に使用されるデフォルトの書式マスク
- ソート基準
- キャラクタ・セット

注意： この環境変数は、Oracle9iAS をインストールする際に自動的に設定されます。Oracle9iAS のインストール後に環境変数を変更する方法の詳細は、[12.2.1.1 項「NLS_LANG 環境変数の定義」](#)を参照してください。

NLS_LANG の構文は、次のとおりです。

NLS_LANG=<language_territory>.<charset>

各値は次のように定義されます。

- language
メッセージ（エラー・メッセージを含む）および月と曜日の名前を表示する言語とその表記規則を指定します。言語を指定しない場合は、その値がデフォルトで米語に設定されます。
- territory
地域を指定し、デフォルトの日付書式、数値に使用される小数点記号、通貨記号、および週番号と曜日番号の計算に使用される表記規則も指定します。地域を指定しない場合は、その値がデフォルトでアメリカに設定されます。

■ charset

データを表示するキャラクタ・セットを指定します。これは、使用する言語とプラットフォームに適合するキャラクタ・セットにする必要があります。この引数は、メッセージの表示に使用するキャラクタ・セットも指定します。

表 12-2 では、NLS_LANG の language、territory、character に一般に使用される値を示します。

表 12-2 一般に使用される NLS_LANG の値

言語	Language_Territory.Character Set
米語	AMERICAN_AMERICA.US7ASCII
アラビア語	ARABIC_UNITED ARAB EMIRATES.AR8ISO8859P6
ポルトガル語 (ブラジル)	BRAZILIAN PORTUGUESE_BRAZIL.WE8DEC
ブルガリア語	BULGARIAN_BULGARIA.CL8ISO8859P5
フランス語 (カナダ)	CANADIAN FRENCH_CANADA.WE8ISO8859P1
カタロニア語	CATALAN_CATALONIA.WE8ISO8859P1
クロアチア語	CROATIAN_CROATIA.EE8ISO8859P2
チェコ語	CZECH_CZECH REPUBLIC.WE8ISO8859P1
デンマーク語	DANISH_DENMARK.WE8ISO8859P1
オランダ語	DUTCH_THE NETHERLANDS.WE8ISO8859P1
エジプト語	ARABIC_UNITED ARAB EMIRATES.AR8ISO8859P6
英語 (アメリカ)	「米語」を参照
英語 (イギリス)	ENGLISH_UNITED KINGDOM.WE8DEC
エストニア語	ESTONIAN_ESTONIA.BLT8MSWIN1257
フィンランド語	FINNISH_FINLAND.WE8ISO8859P1
フランス語	FRENCH_FRANCE.WE8ISO8859P1
ドイツ語	GERMAN_GERMANY.WE8ISO8859P1
ギリシア語	GREEK_GREECE.EL8ISO8859P7
ヘブライ語	HEBREW_ISRAEL.IW8ISO8859P8
ハンガリー語	HUNGARIAN_HUNGARY.EE8ISO8859P2
アイスランド語	ICELANDIC_ICELAND.WE8ISO8859P1
インドネシア語	INDONESIAN_INDONESIA.WE8ISO8859P1

表 12-2 一般に使用される NLS_LANG の値 (続き)

言語	Language_Territory.Character Set
イタリア語	ITALIAN_ITALY.WE8DEC
日本語	JAPANESE_JAPAN.JA16EUC
韓国語	KOREAN_KOREA.KO16KSC5601
スペイン語 (南米)	LATIN AMERICAN SPANISH_AMERICA.WE8DEC
ラトビア語	LATVIAN_LATVIA.NEE8ISO8859P4
リトアニア語	LITHUANIAN_LITHUANIA.NEE8ISO8859P4
スペイン語 (メキシコ) (「 スペイン語 」も参照)	MEXICAN SPANISH_MEXICO.WE8DEC
ノルウェー語	NORWEGIAN_NORWAY.WE8DEC
ポーランド語	POLISH_POLAND.EE8ISO8859P2
ポルトガル語	PORTUGUESE_PORTUGAL.WE8DEC
ルーマニア語	ROMANIAN_ROMANIA.EE8ISO8859P2
ロシア語	RUSSIAN_CIS.RU8PC855
簡体字中国語	SIMPLIFIED CHINISE_CHINA.ZHS16CGB231280
スロバキア語	SLOVAK_SLOVAKIA.EE8ISO8859P2
スペイン語 (「 スペイン語 (メキシコ) 」も参照)	SPANISH_SPAIN.WE8DEC
スウェーデン語	SWEDISH_SWEDEN.WE8DEC
タイ語	THAI_THAILAND.TH8TISASCII
繁体字中国語	TRADITIONAL CHINESE_TAIWAN.ZHT32EUC
トルコ語	TURKISH_TURKEY.WE8ISO8859P9
ウクライナ語	UKRAINIAN_UKRAINE.CL8ISO8859P5
ベトナム語	VIETNAMESE_VIETNAM.VN8VN3

NLS_LANG の設定では、(基本的に) 同じ言語を使用する各国間の地域的な違いを考慮する必要があります。たとえば、フランスで使用されているフランス語で実行する場合は、NLS_LANG 環境変数を次のように設定します。

```
NLS_LANG=FRENCH_FRANCE.WE8ISO8859P1
```

一方、スイスで使用されているフランス語で実行する場合は、NLS_LANG 環境変数を次のように設定します。

```
NLS_LANG=FRENCH_SWITZERLAND.WE8ISO8859P1
```

12.2.1.1 NLS_LANG 環境変数の定義

Windows または UNIX のオペレーティング・システムでは、NLS_LANG 環境変数を他の環境変数と同じ方法で定義します。

12.2.1.1.1 Windows Windows で NLS_LANG 環境変数を定義する手順は、次のとおりです。

4. Windows のレジストリを開きます。

注意： レジストリは、バックアップを作成してから編集してください。

5. 「HKEY_LOCAL_MACHINE」ノードを拡張し、次に「SOFTWARE」ノードを拡張します。
6. 「ORACLE」をクリックして、レジストリ・エディタの右側のパネルに Oracle 環境変数を表示します。
7. NLS_LANG 環境変数をダブルクリックします。
8. 「値」データ・テキスト・ボックスに NLS_LANG の新しい値を入力します。
9. 「OK」をクリックします。

12.2.1.1.2 UNIX UNIX プラットフォームで NLS_LANG 環境変数を定義するには、ORACLE_HOME/bin ディレクトリにあるシェル・スクリプト `rwrun.sh` でその環境変数を設定します。

12.2.1.2 キャラクタ・セット

NLS 環境変数のキャラクタ・セット・コンポーネントでは、各自の環境でデータを表す際に使用されるキャラクタ・セットを指定します。あるキャラクタ・セットを使用するシステムから別のキャラクタ・セットを使用するシステムにデータが転送されると、転送先のシステムでそのデータが処理され、正しく表示されます。ただし、一部の文字がキャラクタ・セットの別のバイナリ値で表される場合があります。

12.2.1.2.1 キャラクタ・セットの設計上の考慮事項 多言語対応アプリケーション、または単一言語対応アプリケーションであっても複数のキャラクタ・セットを使用するアプリケーションを設計する場合は、実行時に最も広く使用されるキャラクタ・セットを決定し、NLS 環境変数をそのキャラクタ・セットに設定してアプリケーションを生成する必要があります。

あるキャラクタ・セットで設計および生成されたアプリケーションを別のキャラクタ・セットで実行すると、パフォーマンスに問題が生じる可能性があります。さらに、生成時のキャラクタ・セットの文字の一部が実行時のキャラクタ・セットに含まれていない場合は、認識されない文字の部分に疑問符が表示されます。

Portable Document Format (PDF) は、マルチバイト・キャラクタ・セットをサポートします。

12.2.1.2.2 Windows プラットフォームでのフォントのエイリアシング 特定のフォントを使用してアプリケーションを作成しても、実行してみると別のフォントが使用されている場合があります。この問題は、多くの場合、西ヨーロッパ圏以外の言語環境で英語フォント (MS Sans Serif や Arial など) を使用すると発生します。この問題が起こる理由は、フォントに関連付けられているキャラクタ・セットと言語環境変数によって指定されたキャラクタ・セットが一致しているかを、Oracle9iAS Reports Services がチェックするためです。2つのキャラクタ・セットが一致しない場合は、Oracle9iAS Reports Services によって、言語環境変数で指定されたキャラクタ・セットが関連付けられている別のフォントに自動的に置換されます。この自動置換によって、データベースから返されるデータがアプリケーション内で正しく表示されます。

注意： 英語フォントを使用して各国語の文字を入力すると、Windows では暗黙的に別のフォントへの関連付けが行われます。

ただし、このような自動置換が行われないようにする場合もあります。自動置換が行われないようにするには、希望するすべてのフォントを、フォント・エイリアス・ファイル (uifont.ali) 内の WE8ISO8859P1 キャラクタ・セットにマッピングします。たとえば、アプリケーションで Arial フォントを使用できないときは、フォント・エイリアス・ファイルに次の行を追加します (このファイルは `ORACLE_HOME¥TOOLS¥COMMON¥` にあります)。

```
ARIAL.....=ARIAL.....WE8ISO8859P1
```

uifont.ali ファイル内の各行の構文は、次のとおりです。

```
<Face>.<Size>.<Style>.<Weight>.<Width>.<CharSet>=<Face>.<Size>.<Style>.<Weight>.<Width>.<CharSet>
```

この例では、すべての ARIAL フォントを、WE8ISO8859P1 キャラクタ・セットでの同じ値にマップするように指定しています。

言語環境変数の詳細は、[12.2 項「NLS の環境変数」](#)を参照してください。

12.2.1.3 言語と地域

キャラクタ・セットによって各言語に必要な個別の文字が使用可能になると同時に、各国の表記規則がサポートされることで、データ項目が正しくローカライズされて表示されます。

指定の言語では、次の要素についてデフォルトの表記規則が決まります。

- サーバー・メッセージの言語
- 月と曜日の名前およびそれらの省略形 (SQL 機能 TO_CHAR と TO_DATE で指定) の言語
- AM、PM、AD および BC を表す記号
- ORDER BY を指定したときの文字データのデフォルト・ソート基準 (ORDER BY が指定されない限り、GROUP BY はバイナリ・ソートを使用)
- 表記の方向 (右から左と左から右)
- 肯定応答と否定応答の文字列

たとえば、言語をフランス語に設定すると、英語による次のメッセージがフランス語に変換されます。

English:

```
ORA-00942: table or view does not exist
FRM-10043: Cannot open file.
```

French:

```
ORA-0092: table ou vue inexistante
FRM-10043: Ouverture de fichier impossible
```

指定された地域によって、次のような日付および数値のデフォルトの書式特性に関する表記規則が決まります。

- 日付書式
- 小数点記号および桁グループ・セパレータ
- 各国通貨記号
- ISO 通貨記号
- 週を開始する曜日
- 貸方と借方の記号
- ISO 週フラグ
- リスト・セパレータ

たとえば、地域をフランスに設定すると、カンマを小数点記号として数値が書式設定されます。

12.2.2 DEVELOPER_NLS_LANG 環境変数と USER_NLS_LANG 環境変数

2 か国語のリソース・ファイルとメッセージ・ファイルを同時に使用する必要がある場合は、次の 2 つの言語環境変数を使用します。これらの環境変数は、Oracle9iAS のインストールが完了した後に使用可能となります。

- DEVELOPER_NLS_LANG
- USER_NLS_LANG

DEVELOPER_NLS_LANG と USER_NLS_LANG の構文は、NLS_LANG 環境変数と同じです。したがって次の構文を使用します。

```
DEVELOPER_NLS_LANG=<language_territory>.<charset>
USER_NLS_LANG=<language_territory>.<charset>
```

これらの環境変数は、次のような状況のときに、NLS_LANG 環境変数のかわりに使用します。

- Reports Builder を英語で使用する一方で、別の言語のアプリケーションを開発する場合。DEVELOPER_NLS_LANG および USER_NLS_LANG 環境変数によって、Reports Builder と Reports Runtime で異なる言語設定を使用できます。
- Oracle9iDS Reports Builder のローカル言語バージョンが用意されていない言語で動作するアプリケーションを作成している場合。

これらの環境変数が設定されない場合は、NLS_LANG のデフォルト値が使用されます。

12.3 JSP または XML ファイルでのキャラクタ・セットの指定

JSP で HTML を生成している場合は、次の構文を使用して JSP ファイルにキャラクタ・セットを追加しなければならない場合があります（ここでは日本語のキャラクタ・セットを指定）。

```
<META http-equiv="Content-Type" content="text/html; charset=shift_jis">
```

さらに、レポートを XML に出力する予定がある場合は、Builder のプロパティ・インスペクタで、次の構文に従って XML Prolog Value プロパティにキャラクタ・セットを含めることができます。

```
<?xml version="1.0" encoding="shift_jis" ?>
```

どちらの場合も、キャラクタ・セットを表す値によって、ホスト環境に指定されているキャラクタ・セットと互換性のあるキャラクタ・セットが呼び出されます。Web で使用されるキャラクタ・セットの値は、NLS_LANG 環境変数で表される値と異なります。表 12-3 では、charset または encoding パラメータで一般に使用される値を示します。

注意： charset と encoding の値には、大文字と小文字の区別がありません。小文字と大文字のどちらでも入力できます。

注意： HTML の生成に使用する予定のキャラクタ・セットを .rdf ファイルに設定するには、Before Report Escape プロパティに次の記述が必要です。

```
charset="text/html; charset=&encoding"
&encoding は、実行時に適切な設定で置き換えられます。
```

表 12-3 charset または encoding パラメータの有効値

言語	有効なキャラクタ・セット
アフリカーンス語	iso-8859-1、windows-1252
アルバニア語	iso-8859-1、windows-1252
アラビア語	iso-8859-6
バスク語	iso-8859-1、windows-1252
ブルガリア語	iso-8859-5
ベラルーシ語	iso-8859-5
カタロニア語	iso-8859-1、windows-1252
クロアチア語	iso-8859-2
チェコ語	iso-8859-2
デンマーク語	iso-8859-1、windows-1252
オランダ語	iso08859-1、windows-1252
英語	iso-8859-1、windows-1252
エスペ란anto語	iso-8859-3（サポートするブラウザは少数）
エストニア語	iso-8859-15

表 12-3 charset または encoding パラメータの有効値 (続き)

言語	有効なキャラクタ・セット
フェロー語	iso-8859-1、windows-1252
フィンランド語	iso-8859-1、windows-1252
フランス語	iso-8859-1、windows-1252
ガリシア語	iso-8859-1、windows-1252
ドイツ語	iso-8859-1、windows-1252
ギリシア語	iso-8859-1
ヘブライ語	iso-8859-8
ハンガリー語	iso-8859-2
アイスランド語	iso-8859-1、windows-1252
イヌイット語	iso-8859-10 (サポートするブラウザは少数)
アイルランド語	iso-8859-1
イタリア語	iso-8859-1
日本語	shift_jis、iso-202-jp、euc-jp
韓国語	euc-kr
ラップ語	iso-8859-10 (サポートするブラウザは少数)
ラトビア語	iso-8859-13、windows-1257
リトアニア語	iso-8859-13、windows-1257
マケドニア語	iso-8859-5
マルタ語	iso-8859-3 (サポートするブラウザは少数)
ノルウェー語	iso-8859-1、windows-1252
ポーランド語	iso-8859-2
ポルトガル語	iso-8859-1、windows-1252
ルーマニア語	iso-8859-2
ロシア語	koi-8-r、iso-8859-5
スコットランド語	iso-8859-1、windows-1252
セルビア語	iso-8859-5
スロバキア語	iso-8859-2

表 12-3 charset または encoding パラメータの有効値（続き）

言語	有効なキャラクタ・セット
スロベニア語	iso-8859-2
スペイン語	iso-8859-1、windows-1252
スウェーデン語	iso-8859-1、windows-1252
トルコ語	iso-8859-9、windows-1254
ウクライナ語	iso-8859-5

12.4 双方向サポート

双方向サポートは、通常の文字表記方向が右から左である中東圏および北アフリカ圏の言語でアプリケーションを設計できるようにする機能です。双方向サポートによって、次の制御が可能になります。

- レイアウト方向。右側にラベルがあり、チェック・ボックスとラジオ・ボタンが正しい位置に配置されている項目の表示を含みます。
- 読む方向。テキストの方向（右から左または左から右）を含みます。
- 整列。原点の左上から右上への切替えを含みます。
- 初期キーボード状態。エンド・ユーザーがフォームへの入力を開始したときに、各言語の文字またはローマ字が自動的に生成されるかどうかを制御します（エンド・ユーザーはこの設定を上書きできます）。

双方向のアプリケーションを設計する際に、NLS の環境変数 DEVELOPER_NLS_LANG および USER_NLS_LANG を NLS_LANG のかわりに使用することができます。たとえば、Windows 環境で米語のインタフェースを使用しながらアラビア語のアプリケーションを開発する場合は、これらの環境変数を次のように設定します。

```
DEVELOPER_NLS_LANG=AMERICAN_AMERICA.AR8MSWIN1256
USER_NLS_LANG=ARABIC_UNITED ARAB EMIRATES.AR8MSWIN1256
```

この例では、DEVELOPER_NLS_LANG 環境変数がアラビア語のキャラクタ・セットを使用していることに注意してください。環境変数の詳細は、[「NLS の環境変数」](#)を参照してください。

12.5 Unicode

Unicode は、1 つのアプリケーションで多言語テキストを表示できるようにするグローバルなキャラクタ・セットです。これによって多国籍企業では、1 つの多言語アプリケーションを開発し、それを世界各国に配布できます。

グローバルな市場では、次のようなキャラクタ・セットが必要です。

- 製品の 1 つの実装ですべての言語に対応でき、しかも世界各国で簡単に実装できる。
- 主要なスクリプトをすべて含む。
- 多言語ユーザーおよび多言語組織をサポートする。
- インターネットを介した全世界的なデータ交換を可能にする。

12.5.1 Unicode のサポート

Oracle9iAS Reports Services では Unicode をサポートしています。Unicode を使用すると、西ヨーロッパ、東ヨーロッパ、中東（双方向表記）などで使用されているシングルバイト言語、および中国語、日本語、韓国語（CJK）などのアジア系マルチバイト言語の両方を、同じアプリケーションで表示できるようになります。

すべての言語を網羅する単一のキャラクタ・セットを使用することにより、各種言語の様々なキャラクタ・セットを用意する必要がなくなります。たとえば、日本語などのマルチバイト言語を表示するには、NLS_LANG 環境変数を次のように設定する必要があります。

```
NLS_LANG=JAPAN_JAPANESE.JA16SJIS
```

ドイツ語などのシングルバイト言語を表示するには、NLS_LANG を次のように設定する必要があります。

```
NLS_LANG=GERMAN_GERMANY.WE8ISO8859P1
```

この方式の明らかなデメリットは、アプリケーションで一度に 1 つのキャラクタ・セットの文字しか表示できない点です。複数のキャラクタ・セットを組み合わせたデータは表示できません。

Unicode キャラクタ・セットを使用する場合は、NLS_LANG のキャラクタ・セット部分に、特定言語のキャラクタ・セットではなく UTF8 を設定します。これによって、異なる言語およびキャラクタ・セットの文字を同時に表示できます。たとえば、画面上に日本語とドイツ語を同時に表示するには、NLS_LANG 変数を次のいずれかに設定する必要があります。

```
NLS_LANG=JAPAN_JAPANESE.UTF8
```

```
NLS_LANG=GERMAN_GERMANY.UTF8
```

Unicode 機能によって、アプリケーション開発者とエンド・ユーザーは、フォームに多言語テキストを表示できるようになります。これには、Unicode を含むデータベースのテキスト、多言語テキスト、グラフィカル・ユーザー・インタフェース (GUI) オブジェクト内のテキスト (ボタン・ラベルなど)、キーボード入力されたテキスト、およびクリップボードのテキストが含まれます。Oracle9iAS Reports Services は、現在 Windows 上で Unicode をサポートしています。

注意： Web 用にアプリケーションを開発する場合は、Unicode を使用できます。これは、Java がブラウザを通じて Unicode をサポートしているためです。

12.5.2 Unicode フォントのサポート

Oracle9iAS Reports Services では、様々な言語のフォントと入力方式について、オペレーティング・システムに依存しています。特定の言語でテキストを入力および表示するには、その言語をサポートするオペレーティング・システムのバージョンを実行している必要があります。フォントのサポートは限られていますが、オペレーティング・システムのフォントのみに制限されているわけではありません。

Windows NT 4.0 は、TrueType Big Font を提供しています。これらのフォントには、多言語テキストの表示や印刷に必要なすべての文字が含まれています。多言語テキストを入力、表示または印刷したときに予期しない文字が出力された場合は、Big Font を使用していない可能性があります。Microsoft 社が Windows NT 4.0 で提供している Big Font は、次のとおりです。

- Arial
- Courier New
- Lucida Console
- Lucida Sans Unicode
- Times New Roman

サード・パーティの Unicode フォントも利用できます。

12.5.3 Unicode サポートの有効化

Unicode サポートを有効にするには、NLS_LANG 環境変数を次のように設定します。

```
NLS_LANG=<language_territory>.UTF8
```

環境変数の詳細は、「[NLS の環境変数](#)」を参照してください。

12.5.4 ALTER SESSION の使用方法

USERID キーワードを通じて Oracle データベースに接続されているか、または Reports Builder の「接続」ダイアログから接続した場合は、SQL コマンド ALTER SESSION を使用して、NLS のデフォルト値の一部を上書きすることができます。たとえば、様々な地域にデータを公開するレポートを、Oracle データベースで作成するとします。この場合、Oracle データベースが通貨フィールドを書式設定する際に使用する通貨記号、3 桁の区切りおよび小数点記号を、ユーザー・パラメータに応じて変える必要があります。これにはいくつかの方法がありますが、その 1 つに、BEFORE REPORT トリガーで Oracle データベース・セッションの NLS_LANG 変数を変更する方法があります。

注意： ALTER SESSION は、トランспортаブルなデータ・ソース間合せ (JDBC や XML など) には適用されません。

注意： ALTER SESSION コマンドの詳細は、『Oracle9i SQL リファレンス』を参照してください。

12.6 アプリケーションの翻訳

Oracle9iAS Reports Services のどのアプリケーションにも、次のような様々なメッセージが使用されています。

- データベースからのエラー・メッセージ
- Oracle9iAS Reports Services で生成されるランタイム・エラー・メッセージ
- アプリケーションの一部として定義されるメッセージまたはボイラープレート・テキスト

このうち最初の 2 つについては、NLS 環境変数を正しく設定し、適切なメッセージ・ファイルを用意することで、メッセージが自動的に翻訳されます。アプリケーションの一部として定義されているメッセージおよびボイラープレート・テキストを翻訳するには、Oracle の翻訳ツールである TranslationHub を使用します。また、コードの文字列には PL/SQL ライブラリを使用すると便利です。

注意： TranslationHub の使用方法は、Oracle9iDS ドキュメント CD および Oracle Technology Network (<http://otn.oracle.com/> 英語サイト) または Oracle Technology Network Japan (<http://otn.oracle.co.jp/> 日本語サイト) を参照してください。

PL/SQL ブロック内の定数テキストは、手動で翻訳する必要があります。これは、このテキストが明確に区切られておらず、多くの場合、変数や文字列から構成されているためです。

このような文字列を翻訳するには、PL/SQL ライブラリを使用して柔軟なメッセージ構造を実装します。

ビルトイン・ルーチン `SRW.MESSAGE` によって、またはトリガーやプロシージャから表示項目にメッセージを割り当てることによってプログラムから表示されるメッセージには、連結可能な PL/SQL ライブラリを使用して、柔軟なメッセージ機能を実装できます。ライブラリはホスト上に格納し、実行時に動的に連結できます。実行時には、検索パスに基づいて、連結されたライブラリを読み込むことができます。たとえば、ライブラリにイタリア語のメッセージのみが格納されているとします。

```
FUNCTION nls_appl_mesg(inexe_no NUMBER)
RETURN CHAR
IS
    msg CHAR(80);
BEGIN
    IF      index_no = 1001 THEN
        msg := 'L' 'impiegato che Voi cercate non esiste...';
    ELSEIF index_no = 1002 THEN
        msg := 'Lo stipendio non puo essere minore di zero.';
    ELSEIF ...
    .
    .
    ELSE
        msg := 'ERRORE: Indice messaggio inesistente.';
    END IF;
    RETURN msg;
END;
```

このようなルーチンは、一般に文字式が使用可能な場所であればどこでも使用できます。たとえば、適切に翻訳されたアプリケーション・メッセージを使用して警告を表示するには、次のコードを組み込みます。

```
Change_Alert_Message('My_Error_Alert', nls_appl_mesg(1001));
n := Show_Alert('My_Error_Alert');
```

アプリケーションを別の言語に変更するには、`nls_appl_mesg` ファンクションを含む PL/SQL ライブラリを、翻訳済みテキストを指定した `nls_appl_mesg` ファンクションを含む同名のライブラリに置き換えるだけです。

第 IV 部

パフォーマンス

第 IV 部では、Oracle9iAS Reports Services 環境の管理、監視およびチューニングに関する情報を記載します。第 IV 部には、次の章があります。

- [第 13 章「Oracle9iAS Reports Services の管理と監視」](#)
- [第 14 章「Oracle9iAS Reports Services のチューニング」](#)

Oracle9iAS Reports Services の管理と監視

Oracle9iAS 付属の Oracle Enterprise Manager (OEM) は、Oracle9iAS Reports Services に対して管理サービスと監視サービスを提供します。

OEM は、次の目的で使用します。

- Reports Server の開始、停止および再起動
- Reports のジョブ・キュー（スケジュールされたジョブ、現行ジョブ、失敗したジョブおよび終了したジョブ）の表示と管理
- サーバー・パフォーマンスの監視
- Reports Server 構成ファイルの表示と変更
- サーバー・クラスタの全メンバーの表示と各メンバーへのリンク

注意： Oracle Enterprise Manager の詳細は、『Oracle9i Application Server 管理者ガイド』を参照してください。

Oracle9iAS をインストールすると、Reports Server が自動的に識別され、それらが OEM に登録されます。手動での操作は、OEM の起動と管理の開始だけです。Oracle9iAS をインストールした後で、使用する環境に Reports Server を追加する場合は、OEM の targets.xml ファイルに新しい Reports Server を手動で追加する必要があります。

この章では、Oracle9iAS Reports Services に関連する OEM の管理機能と監視機能について説明するとともに、OEM の targets.xml ファイルに Reports Server を追加する方法を示します。この章の主な項は、次のとおりです。

- [OEM での Reports Services 情報へのナビゲート](#)
- [Reports Server の開始、停止および再起動](#)
- [Reports ジョブ・キューの表示と管理](#)
- [サーバー・パフォーマンスの監視](#)
- [Reports Server 構成ファイルの表示と変更](#)
- [サーバー・クラスタ・メンバーの表示とメンバーへのリンク](#)
- [OEM への Reports Server の追加](#)

注意： OEM の Reports Server 関連ページには、状況依存のオンライン・ヘルプ・トピックが用意されており、各ページの項目に関する情報を提供します。OEM で「ヘルプ」リンクをクリックすると、ヘルプが表示されます。

13.1 OEM での Reports Services 情報へのナビゲート

OEM で Reports Services 情報にナビゲートする手順は、次のとおりです。

注意： Oracle Enterprise Manager コンソールを起動する前に、まず管理サーバーをインストールして構成する必要があります。その方法は、『Oracle Enterprise Manager 構成ガイド』の第 3 章を参照してください。

1. Oracle Enterprise Manager コンソールを起動します。コンソールを起動する方法の詳細は、Enterprise Manager のドキュメントを参照してください。
2. コンソールで、「Oracle9iAS」ノードをクリックします。
3. Oracle9iAS のホーム・ページで、Reports Server のノードをクリックしてその OEM メイン・ページを開きます。

13.2 Reports Server の開始、停止および再起動

OEM に登録された Reports Server は、OEM を通じて停止、開始および再起動できるようになります。

注意： Reports Server は、Oracle9iAS をインストールする際に自動的に OEM に登録されます。Oracle9iAS をインストールした後で Reports Server を追加する場合は、新しいサーバーを OEM の `targets.xml` ファイルに手動で登録する必要があります。詳細は、[13.7 項「OEM への Reports Server の追加」](#)を参照してください。

Reports Server を開始、停止または再起動する手順は、次のとおりです。

1. OEM で、対象となる Reports Server までナビゲートします。
2. Reports Server のメイン OEM ページで次の操作を行います。
 - サーバーを開始する場合は、「開始」ボタンをクリックします。
 - サーバーを停止する場合は、「停止」ボタンをクリックします。
 - サーバーを再起動する場合は、「再起動」ボタンをクリックします。

これらのボタンは、サーバーの現在の状態に応じて Reports Server のメイン OEM ページに表示されます。

- サーバーが停止している場合は、「開始」ボタンと「停止」ボタンが表示されます。
- サーバーが稼働している場合は、「再起動」ボタンと「停止」ボタンが表示されます。

13.3 Reports ジョブ・キューの表示と管理

OEM には、Reports のジョブ・キューを表示および管理するためのページがあります。各キュー（現行ジョブ、スケジュールされたジョブ、失敗したジョブおよび終了したジョブのキュー）に専用のページが用意されています。一部の情報から、より詳細な情報へとドリルダウンすることができます。たとえば、終了したジョブのページでは、ドリルダウンによって、情報つまりレポート出力の Web ビュー（Reports Server のキャッシュから取得）を追跡できます。

この後の項では、次の操作について説明します。

- [現行ジョブ・キューの表示と管理](#)
- [スケジュールされたジョブ・キューの表示と管理](#)
- [終了したジョブ・キューの表示と管理](#)
- [失敗したジョブ・キューの表示と管理](#)

13.3.1 現行ジョブ・キューの表示と管理

Reports の現行ジョブ・キューには、特定の Reports Server で現在実行されているすべてのジョブが示されます。

OEM では、次の操作が可能です。

- [Reports Server の現行ジョブ・キューの表示](#)
- [現行ジョブの取消し](#)

13.3.1.1 Reports Server の現行ジョブ・キューの表示

現行ジョブ・キューを表示する手順は、次のとおりです。

1. OEM で、対象となる Reports Server までナビゲートします。
2. Reports Server のメイン・ページで、「応答とロード」セクションまでスクロールし、「現行ジョブ」の横の数字をクリックします。

現行ジョブ・キューに現行ジョブがない場合は、リンクが表示されず、空のキューを表示することもできません。

[表 13-1](#) では、現行ジョブ・キューに表示される情報を説明します。

表 13-1 現行ジョブ・キューに表示される情報

項目	説明
選択	特定のジョブを選択するために使用するラジオ・ボタン。現行ジョブ・キューのページでは、ジョブを取り消す際にこの機能が最も役立ちます。取り消すジョブの横にある「選択」ラジオ・ボタンをクリックし、ページ上部の「取消」ボタンをクリックします。
ID	Reports Server がこのジョブに割り当てた一意のジョブ識別子。この番号はサーバーで管理されるため、ユーザーはリセットできません。
ジョブ名	このジョブを実行する際に使用したコマンドラインにジョブ名を指定した場合はその名前が表示され、ジョブ名を指定しなかった場合は、ジョブ要求の "report=" または "module=" パラメータに指定されたジョブの名前が表示されます。
所有者	このジョブを実行しているユーザーの ID。
出力タイプ	実行時にこのジョブに指定された宛先タイプ (destype)。
出力フォーマット	実行時にこのジョブに指定された出力フォーマット (desformat)。
キューイングした時間	この要求がジョブ・キューに登録された日付と時刻。
開始時間	このジョブの実行が開始された日付と時刻。

13.3.1.2 現行ジョブの取消し

現行ジョブを取り消す手順は、次のとおりです。

1. 現行ジョブ・キューのページで、取り消すジョブの横にある「選択」ラジオ・ボタンをクリックします。
2. 「ジョブの取消」ボタンをクリックします。

現在実行されているジョブがない場合は、OEM にこのボタンが表示されません。

ジョブを再実行する場合は、終了したジョブ・キューからそれを実行できます。

13.3.2 スケジュールされたジョブ・キューの表示と管理

Reports のスケジュールされたジョブ・キューには、特定の Reports Server で実行がスケジュールされているすべてのジョブが示されます。

OEM では、次の操作が可能です。

- [Reports Server のスケジュールされたジョブ・キューの表示](#)
- [スケジュールされたジョブの取消し](#)

13.3.2.1 Reports Server のスケジュールされたジョブ・キューの表示

スケジュールされたジョブ・キューを表示する手順は、次のとおりです。

- 1. OEM で、対象となる Reports Server までナビゲートします。
- 2. Reports Server のメイン・ページで、「応答とロード」セクションまでスクロールし、「スケジュールされたジョブ」の横の数字をクリックします。

スケジュールされたジョブ・キューにスケジュールされたジョブがない場合は、リンクが表示されず、空のキューを表示することもできません。

表 13-2 では、スケジュールされたジョブ・キューに表示される情報を説明します。

表 13-2 スケジュールされたジョブ・キューに表示される情報

項目	説明
選択	特定のジョブを選択するために使用するラジオ・ボタン。スケジュールされたジョブ・キューのページでは、ジョブを取り消す際にこの機能が最も役立ちます。取り消すジョブの横にある「選択」ラジオ・ボタンをクリックし、ページ上部の「取消」ボタンをクリックします。
ID	Reports Server がこのジョブに割り当てた一意のジョブ識別子。この番号はサーバーで管理されるため、ユーザーはリセットできません。
ジョブ名	このジョブを実行する際に使用したコマンドラインにジョブ名を指定した場合はその名前が表示され、ジョブ名を指定しなかった場合は、ジョブ要求の "report=" または "module=" パラメータに指定されたジョブの名前が表示されます。
所有者	このジョブの実行をスケジュールしたユーザーの ID。
出力タイプ	このジョブに指定された宛先タイプ (destype)。
出力フォーマット	このジョブに指定された出力フォーマット (desformat)。
キューイングした時間	この要求がジョブ・キューに登録された日付と時刻。
間隔	ジョブが実行される頻度 (毎日、毎月など)。

13.3.2.2 スケジュールされたジョブの取消し

スケジュールされたジョブを取り消す手順は、次のとおりです。

- 1. スケジュールされたジョブ・キューのページで、取り消すジョブの横にある「選択」ラジオ・ボタンをクリックします。
- 2. 「ジョブの取消」ボタンをクリックします。

スケジュールされたジョブがない場合は、OEM にこのボタンが表示されません。

ジョブを再実行する場合は、終了したジョブ・キューからそれを実行できます。

13.3.3 終了したジョブ・キューの表示と管理

Reports の終了したジョブ・キューには、特定の Reports Server で正常に終了したすべてのジョブが示されます。

OEM では、次の操作が可能です。

- [Reports Server の終了したジョブ・キューの表示](#)
- [ジョブのトレース・ファイルの表示](#)
- [キャッシュからの結果の表示](#)
- [終了したジョブの再実行](#)

13.3.3.1 Reports Server の終了したジョブ・キューの表示

終了したジョブ・キューを表示する手順は、次のとおりです。

1. OEM で、対象となる Reports Server までナビゲートします。
2. Reports Server のメイン・ページで、「応答とロード」セクションまでスクロールし、「終了したジョブ」の横の数字をクリックします。

終了したジョブ・キューに終了したジョブがない場合は、リンクが表示されず、空のキューを表示することもできません。

[表 13-3](#) では、終了したジョブ・キューに表示される情報を説明します。

表 13-3 終了したジョブ・キューに表示される情報

項目	説明
選択	<p>特定のジョブを選択するために使用するラジオ・ボタン。終了したジョブ・キューのページでは、ジョブを選択して次の操作を行う際にこの機能が最も役立ちます。</p> <ul style="list-style-type: none">■ ジョブ出力の表示 表示するジョブの横にある「選択」ラジオ・ボタンをクリックし、ページ上部の「結果の表示」ボタンをクリックします。■ ジョブのトレース結果の表示（ランタイム・コマンドにトレース・コマンドを指定した場合） トレース結果を表示するジョブの横にある「選択」ラジオ・ボタンをクリックし、ページ上部の「トレースの表示」ボタンをクリックします。■ ジョブの再実行 再実行するジョブの横にある「選択」ラジオ・ボタンをクリックし、ページ上部の「レポートの再実行」ボタンをクリックします。
ID	Reports Server がこのジョブに割り当てた一意のジョブ識別子。この番号はサーバーで管理されるため、ユーザーはリセットできません。
ジョブ名	このジョブを実行する際に使用したコマンドラインにジョブ名を指定した場合はその名前が表示され、ジョブ名を指定しなかった場合は、ジョブ要求の "report=" または "module=" パラメータに指定されたジョブの名前が表示されます。
所有者	このジョブを実行したユーザーの ID。
出力タイプ	このジョブに指定された宛先タイプ（destype）。
出力フォーマット	このジョブに指定された出力フォーマット（desformat）。
キューイングした時間	この要求がジョブ・キューに登録された日付と時刻。
開始時間	このジョブの実行が開始された日付と時刻。
終了時間	このジョブが完了した日付と時刻。
ステータス	ジョブの終了状態。終了したジョブ・キューでは、ステータスが常に「正常に終了しました」と表示されます。

13.3.3.2 ジョブのトレース・ファイルの表示

ジョブのトレース・ファイルを表示する手順は、次のとおりです。

1. 終了したジョブ・キューのページにある「選択」列で、トレース・ファイルを表示するジョブの横にあるラジオ・ボタンをクリックします。
2. ページ上部の「トレースの表示」ボタンをクリックします。

13.3.3.3 キャッシュからの結果の表示

Reports Server のキャッシュからジョブ結果を表示する手順は、次のとおりです。

1. 終了したジョブ・キューのページにある「選択」列で、表示するジョブの横にあるラジオ・ボタンをクリックします。
2. ページ上部の「結果の表示」ボタンをクリックします。
新しいブラウザ・ウィンドウに結果が表示されます。

13.3.3.4 終了したジョブの再実行

ジョブを再実行する手順は、次のとおりです。

1. 終了したジョブ・キューのページにある「選択」列で、再実行するジョブの横にあるラジオ・ボタンをクリックします。
2. ページ上部の「レポートの再実行」ボタンをクリックします。

13.3.4 失敗したジョブ・キューの表示と管理

Reports の失敗したジョブ・キューには、特定の Reports Server で取り消された、またはエラーで終了したすべてのジョブが示されます。

OEM では、次の操作が可能です。

- [Reports Server の失敗したジョブ・キューの表示](#)
- [失敗したジョブのトレース・ファイルの表示](#)
- [失敗したジョブの再実行](#)

13.3.4.1 Reports Server の失敗したジョブ・キューの表示

失敗したジョブ・キューを表示する手順は、次のとおりです。

1. OEM で、対象となる Reports Server までナビゲートします。
2. Reports Server のメイン・ページで、「応答とロード」セクションまでスクロールし、「失敗したジョブ」の横の数字をクリックします。

失敗したジョブ・キューに失敗したジョブがない場合は、リンクが表示されず、空のキューを表示することもできません。

表 13-4 では、失敗したジョブ・キューに表示される情報を説明します。

表 13-4 失敗したジョブ・キューに表示される情報

項目	説明
選択	特定のジョブを選択するために使用するラジオ・ボタン。失敗したジョブ・キューのページでは、ジョブを選択して次の操作を行う際にこの機能が最も役立ちます。 <ul style="list-style-type: none">■ ジョブのトレース結果の表示（ランタイム・コマンドにトレース・コマンドを指定した場合） トレース結果を表示するジョブの横にある「選択」ラジオ・ボタンをクリックし、ページ上部の「トレースの表示」ボタンをクリックします。■ ジョブの再実行 再実行するジョブの横にある「選択」ラジオ・ボタンをクリックし、ページ上部の「レポートの再実行」ボタンをクリックします。
ID	Reports Server がこのジョブに割り当てた一意のジョブ識別子。この番号はサーバーで管理されるため、ユーザーはリセットできません。
ジョブ名	このジョブを実行する際に使用したコマンドラインにジョブ名を指定した場合はその名前が表示され、ジョブ名を指定しなかった場合は、ジョブ要求の "report=" または "module=" パラメータに指定されたジョブの名前が表示されます。
所有者	このジョブを実行したユーザーの ID。
出力タイプ	このジョブに指定された宛先タイプ（destype）。
出力フォーマット	このジョブに指定された出力フォーマット（desformat）。
キューイングした時間	この要求がジョブ・キューに登録された日付と時刻。
開始時間	このジョブの実行が開始された日付と時刻。
終了時間	このジョブが取り消されたか、またはエラーで終了した日付と時刻。
ステータス	ジョブの終了状態。このステータスで、ジョブがユーザーによって取り消されたか、またはエラーで終了したかを確認できます。ジョブがエラーで終了した場合は、その原因を示す簡単なエラー・メッセージが表示されます。

13.3.4.2 失敗したジョブのトレース・ファイルの表示

失敗したジョブのトレース・ファイルを表示する手順は、次のとおりです。

1. 失敗したジョブ・キューのページにある「選択」列で、トレース・ファイルを表示するジョブの横にあるラジオ・ボタンをクリックします。
2. ページ上部の「トレースの表示」ボタンをクリックします。

13.3.4.3 失敗したジョブの再実行

失敗したジョブを再実行する手順は、次のとおりです。

1. 失敗したジョブ・キューのページにある「選択」列で、再実行するジョブの横にあるラジオ・ボタンをクリックします。
2. ページ上部の「レポートの再実行」ボタンをクリックします。

13.4 サーバー・パフォーマンスの監視

OEMに登録されている各 Reports Server には、サーバーのステータスとパフォーマンスに関する一般情報を要約するホーム・ページが個別に用意されます。パフォーマンスに関する情報を提供するセクションは、「Reports Server」、「パフォーマンス」および「管理」です。

- 「Reports Server」セクションは、さらに「一般」、「構成」、「ステータス」および「応答とロード」に分かれています。
 - 「一般」には、インストールされている Reports Server のバージョン番号と、サーバーの稼働状態（稼働中または停止中）が表示されます。稼働中である場合は、サーバーが開始された日付と時刻も表示されます。このセクションには、サーバーを停止、開始または再起動するためのボタンが表示されます。表示されるボタンは、サーバーの稼働状態によって異なります。
 - 「構成」には、選択した Reports Server のクラスタ名（そのサーバーがクラスタのメンバーである場合）、その構成ファイルに指定されているトレース・オプションとモード、および Reports Server のキューに一度に保持できる最大ジョブ数（現行ジョブ、スケジュールされたジョブ、終了したジョブおよび失敗したジョブの各キューの合計を含む）が表示されます。これらの値は、すべて設定可能です。
 - 「ステータス」には、選択した Reports Server 上で現在アクティブなエンジンの数、選択したサーバーが現在使用しているホスト・マシンの CPU と RAM の量、および選択した Reports Server がクライアントからの要求を処理するために要する平均時間（ミリ秒）が表示されます。
 - 「応答とロード」には、現行ジョブ、失敗したジョブ、終了したジョブおよびスケジュールされたジョブの数に関する情報が表示されます。

- 「パフォーマンス」セクションには、詳細なパフォーマンス情報へのリンクが表示されます。各リンクから、同じパフォーマンス・ページの別のセクションにアクセスできます。
- 「応答メトリック」には、平均応答時間、ジョブ・キュー内のスケジュールされたジョブ、終了したジョブ、現行ジョブおよび失敗したジョブ、およびクラスタ環境内のあるサーバーから別のサーバーに転送されたジョブの数に関する詳細情報が表示されます。
- 「エンジン情報」には、選択した Reports Server で現在実行されているエンジンのタイプと数が示されます。
- 「システム使用メトリック」には、選択した Reports Server が現在使用する CPU の割合および RAM のメガバイト数が表示されます。
- 「管理」セクションには、選択した Reports Server の現行の構成プロパティに関する詳細情報へのリンク、およびサーバーのトレース・ファイルとログ・ファイルのビューへのリンクが表示されます。
 - 「サーバー設定」からは、選択した Reports Server の構成ファイルの編集可能なビューにアクセスできます。このビューでは、ファイルの変更、ファイル構文のチェックおよび変更の保存を実行できます。変更内容は、次にサーバーを再起動した後に有効になります。詳細は、[13.5 項「Reports Server 構成ファイルの表示と変更」](#)を参照してください。第 3 章「[Oracle9iAS Reports Services の構成](#)」も参照してください。
 - 「サーバー・トレース」からは、選択した Reports Server で実行したあらゆるトレースの結果にアクセスできます。Reports Server の構成ファイルに指定されたトレース・オプションを使用するかどうかを指定します。このファイルには、「サーバー設定」リンクからアクセスできます。
 - 「サーバー・ログ」からは、選択した Reports Server の開始時刻や停止時刻などの一般的なサーバー・イベントのログにアクセスできます。

13.5 Reports Server 構成ファイルの表示と変更

OEM で Reports Server の構成ファイルを表示および変更する手順は、次のとおりです。

1. 選択した Reports Server のホーム・ページで、「管理」の下に「サーバー設定」リンクをクリックします。
2. 表示ウィンドウで必要な変更を加えます。
3. 「構文チェック」ボタンをクリックして XML の構文をチェックします。

注意： このボタンをクリックしても、構成要素に対して入力した値は検証されません。たとえば、ある要素がディレクトリ・パスの指定を必要とする場合でも、構文チェックでは指定したパスが正しいかどうかは検証されません。このチェックでは、XML の構文のみが検証されます。

4. 「変更の保存」ボタンをクリックして変更を保存します。

変更内容は、次にサーバーを起動または再起動した後に有効になります。

注意： Reports Server 構成ファイルに指定される属性と値の詳細は、[第 3 章「Oracle9iAS Reports Services の構成」](#)を参照してください。

13.6 サーバー・クラスタ・メンバーの表示とメンバーへのリンク

複数の Reports Server をクラスタ化すると、OEM で各クラスタ・メンバーのホーム・ページに表示される「Other Servers Running in This Cluster」の下にクラスタ化の内容が反映されます。リスト内の各クラスタ・メンバーから、そのメンバーのホーム・ページにアクセスできます。

[表 13-5](#) では、「Other Servers Running in This Cluster」セクションでクラスタ・メンバーごとに表示される情報を説明します。

表 13-5 「Other Servers Running in this Cluster」の下に表示される情報

行	説明
サーバー名	選択した Reports Server が属しているクラスタのメンバーである他の Reports Server の名前。サーバー名をクリックすると、そのクラスタ・メンバーのホーム・ページにジャンプします。
終了したジョブ	リスト内の Reports Server のジョブ・キューに現在入っている終了したジョブの合計数。
現行ジョブ	リスト内の Reports Server のジョブ・キューに入っている現在実行中のジョブの合計数。
スケジュールされたジョブ	リスト内の Reports Server のジョブ・キューに現在入っているスケジュールされたジョブの合計数。
失敗したジョブ	リスト内の Reports Server のジョブのうち、完了前に停止したジョブの合計数。これには、ユーザーが強制終了したジョブや、エラーで終了したジョブが含まれます。
平均応答時間	リスト内の Reports Server がクライアントからの要求を処理するために要する平均時間（ミリ秒）。

13.7 OEM への Reports Server の追加

Oracle9iAS をインストールする際に、Reports Server は自動的に OEM に登録されます。Oracle9iAS をインストールした後で Reports Server を追加する場合に、そのサーバーを OEM で管理するには、サーバーを手動で OEM に登録する必要があります。この項では、OEM ターゲットの登録情報を含んでいる targets.xml ファイルを編集する方法について説明します。

注意： OEM でのターゲットとは、OEM が管理するコンポーネントのことです。たとえば、OEM に登録される Reports Server は OEM ターゲットと見なされます。

ターゲットを OEM に登録する手順は、次のとおりです。

1. 任意の XML エディタで targets.xml ファイルを開きます。

targets.xml ファイルは、Windows と UNIX どちらの場合も、次のディレクトリ・パスにあります。

```
ORACLE_HOME\sysman\emd\targets.xml
```

2. 次の構文を使用して、追加する Reports Server の情報を入力します。

```
<target type="oracle_repserv" name="Reports_Server_name"
  <property name="password" value="password_value" encrypted="false"/>
  <property name="server" value="Reports_Server_name"/>
  <property name="servlet" value="http_URL_to_Reports_Servlet/rwservlet"/>
  <property name="userName" value="default_userid"/>
  <property name="oracleHome" value="ORACLE_HOME"/>
  <property name="host" value="domain_of_host_machine"/>
  <compositeMembership>
    <memberOf type="oracle_ias" name="ias-1" association="null"/>
  </compositeMembership>
</target>
```

次に例を示します。

```
<target type="oracle_repserv" name="rep_disun1813"
  <property name="password" value="tiger" encrypted="false"/>
  <property name="server" value="rep_disun1813"/>
  <property name="servlet"
    value="http://machinename-pc2.us.oracle.com/servlet/rwservlet"/>
  <property name="userName" value="scott"/>
  <property name="oracleHome" value="d:\ora11"/>
  <property name="host" value="dlsun1813.us.oracle.com"/>
  <compositeMembership>
    <memberOf type="oracle_ias" name="ias-1" association="null"/>
  </compositeMembership>
</target>
```

注意： ユーザー名とパスワードは、Reports Services の管理者アカウントのもので、セキュリティで保護されていない環境では、ユーザー名とパスワードが、Reports Server 構成ファイル <server_name>.conf の <identifier> 要素に対応します。セキュリティで保護された環境では、この 2 つが Oracle9iAS Portal または Oracle Internet Directory (OID) で作成された Reports Services の管理者アカウントに対応します。このアカウントは、RW_ADMINISTRATOR グループに属します。

targets.xml ファイルの password エントリの初期設定は、encrypted="false" にする必要があります。その後 OEM によって値が暗号化され、encrypted が "true" にリセットされます。

3. targets.xml ファイルを保存し、OEM を再起動して変更内容を反映させます。

Oracle9iAS Reports Services のチューニング

Oracle9iAS Reports Services では、多数のチューニング・オプションによって、そのサービスとパフォーマンスのレベルを調整できるようになっています。この章では、チューニングの考慮事項について説明するとともに、サーバー環境をユーザー・ベースの要件に合わせて最適化するための方法を示します。この章には、次の項があります。

- [インプロセス・サーバーの使用方法](#)
- [Reports Engine のチューニング](#)
- [複数サーバーのクラスタ化](#)
- [キャッシュ方法の最適化](#)
- [パフォーマンスの監視](#)

14.1 インプロセス・サーバーの使用法

インプロセス・サーバーは Oracle9iAS Reports Services のコンポーネントで、Reports Servlet (rwservlet) 内にあります。このサーバーは、スタンドアロン・サーバーほどメンテナンスを必要としません。これは、インプロセス・サーバーがスタンドアロン・サーバーとは違って、クライアントから Reports Servlet または JSP ベース・レポートの URL を通じて最初に要求を受け取る際に自動的に起動されるためです。また、インプロセス・サーバーを使用するとプロセス間の通信が削減されるので、パフォーマンスが向上します。

Reports Server を Reports Servlet と同じプロセス内で実行するには、Reports Servlet 構成ファイル (rwservlet.properties) に SERVER_IN_PROCESS パラメータを指定します。

このパラメータの構文は、次のとおりです。

```
SERVER_IN_PROCESS=yes
```

Reports Server を Reports Servlet と同じプロセス内で実行しない場合は、*no* (デフォルト値) を入力します。

Reports Servlet 構成ファイルは、Windows と UNIX どちらの場合も、次のパスにあります。

```
%ORACLE_HOME%\reports\conf\rwservlet.properties
```

14.2 Reports Engine のチューニング

Reports Engine には、パフォーマンスのチューニングに関する多数の構成パラメータがあります。

- [initEngine](#)
- [maxEngine](#)
- [minEngine](#)
- [engLife](#)
- [maxIdle](#)
- [callBackTimeOut](#)

これらのパラメータは、Reports Server 構成ファイル <server_name>.conf に含まれる engine 要素の属性です。このファイルは、Windows と UNIX どちらの場合も次のパスにあります。

```
%ORACLE_HOME%\reports\conf\<server_name>.conf
```

注意： Reports Server 構成ファイルの詳細は、[第 3 章「Oracle9iAS Reports Services の構成」](#) を参照してください。

engine 要素の構文は、次のとおりです。

```
<engine id="rwEng" class="oracle.reports.engine.EngineImpl" initEngine="1"
maxEngine="5" minEngine="1" engLife="50" maxIdle="15" callbackTimeOut="10000">
  <property name="sourceDir" value="D:\orawin\reports\server\cache"/>
  <property name="tempDir" value="D:\orawin\reports\server\temp"/>
</engine>
```

この後の項では、*engine* 要素の関連属性を個別に説明します。ここでは、これらの属性がどのように連動するかも説明します。たとえば、*initEngine* 属性は、*maxEngine* および *minEngine* と連動して、サーバーの稼働中にユーザーが常に利用できる Reports Engine の数を設定します。これらの属性は、*engLife* の影響も受けます。この属性は、エンジンがジョブをいくつ処理したら停止されるかを指定します。また、大量の処理が行われる状況でエンジンが起動後に停止されないようにする場合は、*engLife* に比較的大きな値を設定できます。

これらのどの属性にも最適な設定はありません。最良の設定は、要求負荷の程度や、要求負荷が 1 日を通して一定か不安定かなどの要因によって異なります。

この後の項では、要求負荷が一定または変動する場合に大きな属性値または小さな属性値を指定する利点、および応答時間が受ける影響を考察します。

14.2.1 initEngine

initEngine 属性は、Reports Server が初期化の際に起動するエンジンの数を定義します。この属性のデフォルト値は 1 です。*initEngine* 属性は、サーバーが最初に起動されるときのみ影響します。

Reports Server 構成ファイルにおける *initEngine* の構文は、次のとおりです。

```
initEngine="1"
```

サーバー起動時に初期化するエンジンの数が多いほど、Reports Server の起動にも時間がかかり、それらのエンジンを維持するために必要なメモリーの量も増えます。要求負荷が一定であり、かつユーザー要求の数が多い場合は、*initEngine* の値を大きくする方が有効と考えられます。エンジンがすぐ使える状態にあることで得られるメリットは、メモリーの使用量が増えるというデメリットに勝ります。反対に、要求を受け取ったときにエンジンを利用できないと、エンジンを起動する時間が応答のオーバーヘッドに追加されてしまいます。

14.2.2 maxEngine

maxEngine 属性は、あるタイプのエンジンをサーバー上で同時にいくつまで実行できるかを指定します。要求の数が *maxEngine* に指定した数を上回った場合は、あふれた要求が Reports Queue に送信され、エンジンが利用可能になった時点で実行されます。デフォルト値は 1 です。

Reports Server 構成ファイルにおける *maxEngine* の構文は、次のとおりです。

```
maxEngine="4"
```

14.2.3 minEngine

minEngine 属性は、あるタイプのエンジンが、*maxIdle* (アイドル時間が *x* 分を経過した場合にエンジンを停止するように指定する属性) の値に関係なく、常にいくつ実行されている必要があるかを指定します。デフォルト値は 0 です。

Reports Server 構成ファイルにおける *minEngine* の構文は、次のとおりです。

```
minEngine="0"
```

要求の数が少ない環境、または応答時間が最適でないことに対するユーザー・ベースの許容度が高い環境では、デフォルト値の使用がメモリーを保護する点で有効です (不要なエンジンによってメモリーが占有されなくなるため)。要求負荷が一定で、かつ要求量が多い環境では、値を大きくすると応答が速くなります。これは、要求を受け取ったときにエンジンがすでに稼働しているので、エンジンを起動する時間がかからないためです。

注意： *minEngine* に入力する値は、*maxEngine* または *initEngine* に指定した値以下である必要があります。*minEngine* に、*maxEngine* または *initEngine* より大きな値を指定するとエラーになります。

14.2.4 engLife

engLife 属性は、エンジンが停止されるまでに、1 つのエンジンが実行できるジョブの数を指定します。そして、必要に応じて、新しいエンジンが起動されます。この機能は、メモリー構造をクリーン・アップするために用意されています。デフォルト値は 50 です。

Reports Server 構成ファイルにおける *engLife* の構文は、次のとおりです。

```
engLife="120"
```

小さな値を指定すると、エンジンが早めに停止され、その結果メモリーが解放されます。これによって、リソースの再利用も可能となります。大きな値を指定すると、エンジンが長い間稼働状態になります。これは、要求量が終日多く、メモリーの使用量があまり問題とならない環境で有効です。

14.2.5 maxIdle

maxIdle 属性は、現行のエンジン数が *minEngine* に指定した値を上回る場合に、エンジンが停止されるまでの許容アイドル時間 (分) を指定します。

たとえば *minEngine* が 0、*maxIdle* が 30 であり、1 つのエンジンが実行中の場合、このエンジンは 30 分間使用されないと停止します。

デフォルト値は 30 (分) です。

Reports Server 構成ファイルにおける *maxIdle* の構文は、次のとおりです。

```
maxIdle="30"
```

この値を指定するときは、エンジンを起動する時間を省く必要性と、(アイドル状態のエンジンによる) メモリーの無駄な使用をなくす必要性のバランスをとる必要があります。

14.2.6 callBackTimeOut

callBackTimeOut 属性は、サーバーがエンジンを呼び出してからエンジンがサーバーに応答するまでの許容待ち時間（ミリ秒）を指定します。デフォルト値は 60000（ミリ秒）です。通信が確立する前に *callBackTimeOut* 値を超えた場合は、例外が発生し、エラー・メッセージが表示されて処理が停止します。

Reports Server 構成ファイルにおける *callBackTimeOut* の構文は、次のとおりです。

```
callBackTimeOut="60000"
```

サーバー・ホスト・マシンが高速である場合は、この数値を下げておかまいません。マシンの処理速度が遅い場合は、この数値を高くして、サーバーとエンジンが通信を確立できるだけの十分な時間を確保します。この値は、ユーザー・ベースの要件よりもハードウェアの能力に大きく依存します。

14.3 複数サーバーのクラスタ化

クラスタとは、複数のサーバーを 1 つのコミュニティに仮想的にグループ化したものです。このコミュニティの目的は、要求処理をクラスタの各メンバーに効率的に分散することです。Oracle9iAS Reports Services でのクラスタ化は、マスター / スレーブ型ではなく、ピアレベルで行われます。ピアレベルのクラスタ化では、受信要求の分散と処理に対して、クラスタの全メンバーが同等の役割を担います。このため、あるメンバーが停止しても、残りのメンバーが要求負荷の管理を続行します。1 台のマシンの障害が原因でシステム全体がダウンする単一の障害箇所はありません。

クラスタ化のメリットは、複数マシンの複数 CPU をプーリングすることによる処理能力の向上、フェイルセーフ環境、および要求の分散による応答速度の向上です。

各クラスタ・メンバー・マシンは、各サーバー・メンバー上でのレポートの実行が可能になるよう、ほぼ同様に構成する必要があります。つまり、構成ファイルの内容はほぼ同じになります。同じ設定にする必要があるかどうかは、ジョブ関連の設定か、マシン関連の設定かで区別します。ジョブ関連の設定は、すべてのクラスタ・メンバーで同じにする必要があります。ジョブ関連の設定には、セキュリティ、データ・ソースおよび宛先タイプに関する設定などがあります。マシン関連の設定には、*maxEngine*、*minEngine*、*maxIdle*、*initEngine* などの属性があり、これらはメンバーごとに異なる値を設定できます。

また、クラスタ・メンバーは、次の事項に従う必要があります。

- サーバー関連の環境変数には、同じ値を設定します。
- TNS 設定では、同じデータベースを同じ方法で指定します。

複数のサーバーをそれぞれ異なる言語でセットアップする必要がある場合に、それらをクラスタ化することは通常ありません。また、ジョブ関連の構成設定が異なるマシンがあると、クラスタ化の利点を享受できません。

同じクラスタのメンバーとなるサーバーは、クラスタ名を共有し（各サーバー名に追加されます）、同じ公開鍵および秘密鍵を設定する必要があります。

注意： クラスタにサーバーを登録する方法の詳細は、[第 3 章「Oracle9iAS Reports Services の構成」](#)を参照してください。

14.4 キャッシュ方法の最適化

キャッシュは、最近アクセスされたデータの一時的な記憶領域で、データをその本来の記憶領域よりも速く返すことができます。キャッシュからデータをフェッチすると、本来の記憶領域にアクセスして検索する時間が節約され、ネットワークに余分な通信負荷がかからなくなります。

Oracle9iAS Reports Services では、次の 3 つの方法を使用して、キャッシュを最大限に活用できます。

- Reports Server 構成ファイルで Reports キャッシュを設定します。
- コマンドラインを通じて、特定のレポート出力を再利用できる期間およびそのレポートがキャッシュから削除されるタイミングを定義する規則を指定できます。
- JSP ファイルで、JSP の Web Object Cache を利用する `ojsp` タグを使用します。

これらの使用方法については、次の各項で説明します。

- [Reports Server 構成ファイルでのキャッシュの設定](#)
- [コマンドラインでのキャッシュ関連オプションの指定](#)
- [JSP でのキャッシュ・オプションの設定](#)

14.4.1 Reports Server 構成ファイルでのキャッシュの設定

Reports Server 構成ファイル (%ORACLE_HOME%\reports\conf\<server_name>.conf) には、サーバーのキャッシュおよびキャッシュのサイズと位置を定義する Java クラスを指定する *cache* 要素が含まれています。

Reports Server 付属のデフォルト Java クラスを使用する場合の *cache* 要素の構文は、次のとおりです。

```
<cache class="oracle.reports.cache.RWCache">
  <property name="cacheSize" value="50"/>
  <property name="cacheDir" value="%ORACLE_HOME%\reports\server\cache"/>
</cache>
```

コマンドライン・オプション **TOLERANCE** および **EXPIRATION** をともに指定すると、Reports Server のキャッシュに最新のレポート結果が格納され、後続の同じレポート要求に対してそれらの結果が提供されるので、要求の応答時間が速くなります。応答が速くなるのは、データベースから最新データをフェッチしてレポートの表示バージョンを作成するより、キャッシュから完成したレポートをフェッチする方が速いためです。

チューニングに関連するプロパティは、*cacheSize* です。この値には、Reports キャッシュ専用として確保するディスク領域のメガバイト数を指定します。デフォルト値は 50 (メガバイト) です。

キャッシュ用に確保したディスク領域は、キャッシュ専用であり、他の目的には使用できません。Reports Server では、この領域の状況を常に把握して保護します。**EXPIRATION** コマンドでレポートに設定されている制約を除いて、Reports キャッシュは先入れ先出し (FIFO) の原理で機能します。つまり、キャッシュがいっぱいになるとキャッシュ内の最も古いデータが入ってきたデータで上書きされます。

値を大きくするとより多くのデータを格納できますが、検索にかかる時間は長くなります。値を小さくすると、キャッシュ・ヒットは速くなりますが、十分な量のデータを手元に残しておけないので、同じ要求を実行する頻度が高まります。もう 1 つの考慮事項は、汎用のディスク領域から無理なく取り除くことのできるディスク領域の量です。*cacheSize* に指定する値は、これらの考慮事項のバランスをとり、各自のハードウェア・リソースおよびユーザー・ベースの要件にとって最適な結果をもたらすものでなければなりません。

14.4.2 コマンドラインでのキャッシュ関連オプションの指定

Oracle9iAS Reports Services には、キャッシュに格納されている特定のレポートを同様の要求に対して使用できる期間を定義するコマンドライン・オプション ([TOLERANCE](#))、およびレポートがキャッシュから削除されるまでの期間を定義するコマンドライン・オプション ([EXPIRATION](#)) があります。キャッシュを効果的に使用するには、[TOLERANCE](#) と [EXPIRATION](#) が不可欠です。これらが定義されていないと、キャッシュされたレポート結果を再利用する機会が大幅に減少します。

注意： 実行時に [TOLERANCE](#) または [EXPIRATION](#) が指定されていない場合（またはそれぞれの値が 0 である場合）は、[Reports](#) の現行ジョブ・キューに含まれている要求と一致するジョブ出力のみが、キャッシュから再利用されます。

14.4.2.1 TOLERANCE

[TOLERANCE](#) は、レポートのキャッシュされた出力の再利用に対する最大許容時間を設定します。これらの出力は、重複したジョブが検出された場合に使用されます。レポートについて許容時間を設定すると、重複したジョブが見つかった場合の処理時間が短縮されます。

[TOLERANCE](#) は、[rwclient](#)、[rwservlet](#)、[rwcgi](#) の各コマンドで使用できます。

[TOLERANCE](#) が指定されていないときは、キャッシュに重複レポートが見つかった場合でも、[Oracle9iAS Reports Services](#) はそのレポートを再実行します。

同一ジョブが送信されたときにレポートが処理中（つまり現行ジョブ・キューに入っているとき）であった場合は、[TOLERANCE](#) が設定されていない場合やゼロに設定されている場合でも、[Oracle9iAS Reports Services](#) は現在実行中のジョブの出力を再利用します。

[TOLERANCE](#) コマンドの構文は、次のとおりです。

```
TOLERANCE=[number {unit}] or [{Mon DD, YYYY} hh:mi:ss AM/PM {timezone}]
```

number には、0 以上の任意の整数を指定できます。*unit* には、分 (s)、時間 (s) または日 (s) を指定できます。デフォルトの単位は分です。日付はオプションです。日付を指定しない場合は、本日の日付が使用されます。*timezone* もオプションです。タイムゾーンを指定しない場合は、[Reports Server](#) のタイムゾーンが使用されます。

14.4.2.2 EXPIRATION

[EXPIRATION](#) は、レポート出力が削除されるまでキャッシュにとどまることのできる期間を定義します。

[EXPIRATION](#) は、[rwclient](#)、[rwservlet](#)、[rwcgi](#) の各コマンドで使用できます。

同一ジョブが送信されたときにレポートが現行ジョブ・キューに入っていた場合は、[EXPIRATION](#) が設定されていない場合やゼロに設定されている場合でも、サーバーは現在実行中のジョブの出力を再利用します。

EXPIRATION コマンドの構文は、次のとおりです。

```
EXPIRATION=[number {unit}] or [{Month DD, YYYY} hh:mi:ss am/pm {timezone}]
```

number には、0 以上の任意の整数を指定できます。*unit* には、分 (s)、時間 (s) または日 (s) を指定できます。デフォルトの単位は分です。日付はオプションです。日付を指定しない場合は、本日の日付が使用されます。*timezone* もオプションです。タイムゾーンを指定しない場合は、Reports Server のタイムゾーンが使用されます。

14.4.3 JSP でのキャッシュ・オプションの設定

Oracle9iAS Components for Java 2 Enterprise Edition (OC4J) の Web Object Cache は、JSP やサーブレットなどの動的な Web ページで生成された部分的な結果や中間結果を、Java で記述された Web アプリケーションが取得、格納、再利用、後処理およびメンテナンスできるようにするメカニズムです。プログラミング・インタフェースとして、タグ・ライブラリ (JSP で使用) および Java API (サーブレットで使用) が提供されています。

この項では、キャッシュ関連の `ojsp` タグについて説明するとともに、これらのタグの使用方法 (使用環境の構成方法も含む) が記載されたドキュメントを紹介します。

Web Object Cache のタグ・ライブラリは、JSP のカスタム・タグ機能を使用する、Web Object Cache API の便利なラッパーです。JSP では、カスタム・タグを使用してキャッシュを制御します。この API は、基礎となるタグ・ハンドラ・クラスを通じて呼び出されます。

これらのタグは、`ojsp` 接頭辞を使用することから "`ojsp`" タグと呼ばれていますが、JSP ファイルの `taglib` ディレクティブでは、必要な接頭辞をどれでも指定できます。

JSP ファイルには、これらのタグを 1 つ以上指定できます。キャッシュするデータ (キャッシュ・ブロック) は、開始タグと終了タグで囲まれます。複数のキャッシュ・ブロックは、一意の `name` 属性によって互いに区別されます。

表 14-1 では、OC4J で提供されているキャッシュ・タグを説明します。

表 14-1 キャッシュ・タグの説明

タグ	説明
<code>ojsp:cache</code>	HTML または XML フラグメントの一般的なキャラクターベースのキャッシングに使用されます。
<code>ojsp:cacheXMLObj</code>	XML オブジェクトのキャッシングに使用されます。このタグのパラメータは、 <code>cache</code> タグ・パラメータのスーパーセットを構成します。XML ドキュメントを後処理する場合は、Web Object Cache が特に役立ちます。
<code>ojsp:useCacheObj</code>	シリアル化可能な Java オブジェクトの一般的なキャッシングに使用されます。一部のセマンティックと構文は、標準の <code>jsp:useBean</code> タグを模倣しています。

表 14-1 キャッシュ・タグの説明

タグ	説明
ojsp:cacheInclude	このタグは、キャッシュ・タグ <code>cache</code> および <code>cacheXMLObj</code> の機能を、標準の <code>JSP include</code> タグに結合したものです。
ojsp:invalidateCache	キャッシュ・ブロックをプログラム・ロジックから明示的に無効にする場合に使用されます。 <code>ojsp:invalidateCache</code> タグを他のキャッシュ・タグの <code>invalidateCache</code> 属性と混同しないようにしてください。この属性はより限定的で、あらかじめ存在するキャッシュ・オブジェクトを無効にする場合に使用されます。

OC4J の Web Object Cache の詳細は、Oracle Technology Network (<http://otn.oracle.com/> 英語サイト) または Oracle Technology Network Japan (<http://otn.oracle.co.jp/> 日本語サイト) を参照してください。参考になるソースの 1 つが、『Oracle9iAS Containers for J2EE JSP タグ・ライブラリおよびユーティリティ・リファレンス』です。このドキュメントには、タグの構文と例、および `ojsp:cache` タグを使用する環境の構成に関する情報が記載されています。

14.5 パフォーマンスの監視

Oracle9iAS Reports Services には、パフォーマンスを監視する様々な手段が提供されているので、追加のチューニングによってメリットを得られそうな領域が特定できます。サーバー・パフォーマンスを監視する手段として最も頻繁に使用されるものは、次のとおりです。

- Oracle Enterprise Manager の Reports Services 関連ページ。詳細は、[第 13 章「Oracle9iAS Reports Services の管理と監視」](#) を参照してください。
- Oracle Trace 機能。この機能は、Reports Server 構成ファイル、Reports Servlet 構成ファイル、Oracle Enterprise Manager (OEM) の Reports Server 関連ページおよびトレース関連のコマンドライン・オプションから利用できます。
- SHOWJOBS コマンドライン・オプション。
- RW_SERVER_QUEUE 表。これは Reports Server のジョブ・キューに別のウィンドウ (OEM から利用できるウィンドウ以外の) を開きます。

サーバー・パフォーマンスの監視について、次の各項で説明します。

- [Oracle Trace によるパフォーマンスの監視](#)
- [SHOWJOBS コマンド・キーワード](#)
- [RW_SERVER_QUEUE 表へのアクセス](#)
- [キュー・アクティビティによるデータベースの更新](#)

14.5.1 Oracle Trace によるパフォーマンスの監視

Oracle Trace は、SQL の解析、実行、フェッチの統計、待機の統計などの、パフォーマンスとリソース使用率のデータを収集するツールです。問題を抱えている場合でも、あるいは単にチューニングの可能性を調べるためにサーバー・パフォーマンスを評価する場合でも、Oracle Trace を使用してサーバー・パフォーマンスに関する情報を収集し、それを使用してサーバー構成に関する意志決定を報告したり、テクニカル・サポート担当者に最新のサーバー・アクティビティを通知したりできます。

この項では、Trace の概要、つまりトレース機能を利用できる Oracle9iAS Reports Services のコンポーネントに関する情報を提供するとともに、簡単にアクセスできる追加情報のソースを紹介します。

注意： Oracle Trace については、このマニュアルの他の項でかなり広範囲にわたって説明しています。この項では、その情報の記載箇所を示します。

14.5.1.1 Trace の概要

rwbuilder または rwrundl によって生成したトレース・ファイルには、Builder セッション全体のトレースが記録されます。トレース・オプションは、コマンドラインまたは Reports Builder 構成ファイル（%ORACLE_HOME%\reports\conf\rwbuilder.conf）から指定できます。

たとえば、コマンドラインからは次のように指定できます。

```
tracefile=trace.trc traceopts=trace_all tracemode=trace_replace
```

トレース・ファイルの位置は、現在作業中のディレクトリに対する相対位置か、または絶対位置（そのように指定された場合）です。

Reports Builder 構成ファイルでは、次のように指定できます。

```
<trace traceFile="trace.trc" traceOpts="trace_all" traceMode="trace_replace"/>
```

トレース・ファイルの位置は、サーバー・ログ・ディレクトリ（%ORACLE_HOME%\reports\logs）に対する相対位置か、または絶対位置（そのように指定された場合）です。トレース・ファイルを指定しない場合は、デフォルトのトレース・ファイル名として、サーバーの logs ディレクトリにある <hostname>-rwbuilder.trc が使用されます。

コマンドラインのトレース・オプションは、Reports Builder 構成ファイルに指定されたトレース・オプションより優先されます。

Reports Server にトレース・ファイルを生成すると、Reports Server と Reports Engine に対して別々にトレース・ファイルが生成されます。サーバーのトレース・オプションは、サーバー構成ファイル（%ORACLE_HOME%\reports\conf\<server_name>.conf）に指定します。

次に例を示します。

```
<trace traceFile="trace.trc" traceOpts="trace_all" traceMode="trace_replace"/>
```

サーバー・トレースでのトレース・ファイルの位置は、サーバーの logs ディレクトリに対する相対位置です。トレース・ファイル名を指定しない場合は、デフォルトのサーバー・トレース・ファイル名として `<server_name>-<engine_name><engine_number>.trc` が使用されます。

サーブレット・トレースのトレース・オプションは、`rwsvrlet` 構成ファイル (`%ORACLE_HOME%\reports\conf\rwsvrlet.properties`) から指定できます。次に例を示します。

```
TRACEOPTS=TRACE_ALL  
TRACEFILE=rwsvrlet.trc  
TRACEMODE=TRACE_REPLACE
```

サーブレット・トレースでのトレース・ファイルの位置は、サーバー・ログ・ディレクトリ (`%ORACLE_HOME%\reports\logs`) に対する相対位置です。

14.5.1.2 トレース情報のその他のソース

Oracle9iAS Reports Services では、多数の方法でトレースを設定し、トレース結果を表示できます。

- デフォルトのトレース値は、Reports Server 構成ファイル (`%ORACLE_HOME%\reports\conf\server_name.conf`) で設定できます。これについては、[第 3 章の 3.2.1.11 項「trace」](#) で説明しています。
- Reports Servlet および JSP に関するランタイム情報を追跡し、ログに記録する場合は、サーブレット構成ファイル (`ORACLE_HOME%\reports\conf\rwsvrlet.properties`) で `TRACEOPTS` パラメータを使用します。これについては、[3.3.5 項「Reports Servlet と JSP のトレース・オプションの設定」](#) で説明しています。
- Oracle Enterprise Manager を使用して、トレース結果をサーバー全体またはジョブごとに表示します。これについては、[第 13 章「Oracle9iAS Reports Services の管理と監視」](#) で説明しています。
- ランタイム・コマンドラインで、個別のジョブに必要なトレース・オプションを指定します。これについては、[付録 A の A.4.101 項「TRACEOPTS」](#) 以降で説明しています。

注意： URL からランタイム・コマンドを定義する方法は、[第 8 章「レポート要求の実行」](#) で説明しています。

14.5.2 SHOWJOBS コマンド・キーワード

showjobs コマンド・キーワードは、Reports Server のキュー・ステータスの Web ビューを表示する際に便利です。このキーワードは、rwservlet コマンドでのみ使用します。このコマンドでは次の構文を使用します。

`Reports_URL/rwservlet/showjobs?server=server_name&statusformat=desired_format`

表 14-2 では、SHOWJOBS のパラメータの有効値を説明します。

表 14-2 SHOWJOBS のパラメータの有効値

パラメータ	有効値	説明
server	server_name	ジョブ・キュー内の表示したいジョブを処理した Reports Server の名前。 必須。ただし、サーブレット構成ファイル（ <code>%ORACLE_HOME%\reports\conf\rwservlet.properties</code> ）でデフォルト・サーバーを指定した場合は、このパラメータを省略できます。その場合、showjobs はデフォルトを使用します。
statusformat	html xml xmldtd	デフォルト : html <ul style="list-style-type: none">■ 現行ジョブ、過去のジョブおよびスケジュールされたジョブのキューの HTML バージョンをブラウザに表示するには、html を使用します。これによって、これらのキューの外観上最も理解しやすいバージョンを表示できます。■ 現行ジョブ、過去のジョブおよびスケジュールされたジョブのキューの XML バージョンをブラウザに表示するには、xml を使用します。これによって、他のアプリケーションが使用できるジョブ・ステータス情報をプログラムで渡す際に便利な出力が得られます。■ XML にインライン・データ型ディクショナリ (DTD) を加えたジョブ・キューのバージョンをブラウザに表示するには、xmldtd を使用します。この場合も、他のアプリケーションが使用できるジョブ・ステータス情報をプログラムで渡す際に便利な出力が得られます。

14.5.3 RW_SERVER_QUEUE 表へのアクセス

レポート処理を集中管理に移行すると、ユーザーから一定の動作情報を要求される場合があります。たとえば次のような場合です。

- ユーザーが自分の送信したレポートのステータスを要求する場合。
- Oracle9iAS Reports Server 上で同時に作業するユーザーの数を管理者が要求する場合。
これは、環境のサイズ調整およびライセンス準拠の確認に便利です。

Oracle9iAS Reports Server では、ジョブ要求が送信されるたびに現行のレポート・キューをデータベースに転送することで、これらの要求の両方に対応できるようになっています。この情報は、RW_SERVER_QUEUE 表に挿入されます。この表には次のデータが含まれます。

- ジョブの名前
- ジョブの送信者
- 選択された出力フォーマット
- ジョブの現行ステータス
- ジョブがキューに入れられ、開始され、完了した日付と時刻

表 14-3 では、RW_SERVER_QUEUE 表に含まれる情報を説明します。

表 14-3 RW_SERVER_QUEUE 表の構造

列名	説明
JOB_TYPE	リスト内のジョブが CURRENT、PAST または SCHEDULED のどれであるかを示します。
JOB_ID	システムが生成したジョブ識別番号。
JOB_NAME	送信されたジョブの名前 (JOBNAME に値が指定されていない場合はファイル名)。
STATUS_CODE	ジョブの現行ステータス。ステータス・コードの詳細は、表 14-4 「ジョブ送信のステータス・コード」を参照してください。
STATUS_MESSAGE	ステータス・コードに関連するメッセージ・テキスト全体 (レポートが終了した場合は Oracle9iAS のエラー・メッセージを含む)。
COMMAND_LINE	このジョブを送信するために提供されたコマンドライン全体。
OWNER	ジョブを送信したユーザー。Web では、Web サーバーを所有する OS ユーザーがデフォルト・ユーザーとなります。
DESTYPE	レポート出力のフォーマット。
DESNAME	Oracle9iAS Reports Server のキャッシュに保存されない場合のレポート出力の名前。
SERVER	レポートの送信先となった Oracle9iAS Reports Server。

表 14-3 RW_SERVER_QUEUE 表の構造 (続き)

列名	説明
QUEUED	ジョブ送信が特定の Oracle9iAS Reports Server で受信され、キューに入られた日付と時刻。
STARTED	ジョブ送信が実行された日付と時刻。
FINISHED	送信されたジョブが完了した日付と時刻。
LAST_RUN	スケジュールされたジョブが最後に実行された日付と時刻。
NEXT_RUN	スケジュールされたジョブが次に実行される日付と時刻。
REPEAT_INTERVAL	ジョブを実行する頻度。
REPEAT_PATTERN	繰り返しパターン (毎分、毎時間、毎日など)。
CACHE_HIT	ジョブ結果が、ジョブを実行して得たものではなく、キャッシュからフェッチされたかどうかを示します。
CACHE_KEY	要求をキャッシュ済みの結果と比較する際に使用されるキャッシュ・キー。このキーは、ジョブが実行された時刻に関係なくレポート出力結果を一意に特定する文字列です。たとえば、2つの要求が同じキーを持っている場合は、両方を同時に実行した場合に同じ出力が生成されることを意味します。ただし、出力の用途が同じとは限りません (たとえば電子メールへの送信やファイルへの保存など)。

表 14-4 ジョブ送信のステータス・コード

ステータス・コード	定義済みの PL/SQL 定数	ステータス・コードの説明
0	UNKNOWN	このようなステータスはない。
1	ENQUEUED	ジョブがキューで待機している。
2	OPENING	サーバーがレポート定義を開いている。
3	RUNNING	レポートが現在実行されている。
4	FINISHED	ジョブの送信が正常に完了した。
5	TERMINATED_W_ERR	ジョブがエラーで終了した。
6	CRASHED	ジョブの実行中にエンジンがクラッシュした。
7	CANCELED	ユーザーからの要求によりジョブが取り消された。
8	SERVER_SHUTDOWN	Oracle9iAS Reports Server が停止されたため、ジョブが取り消された。

表 14-4 ジョブ送信のステータス・コード（続き）

ステータス・コード	定義済みの PL/SQL 定数	ステータス・コードの説明
9	WILL_RETRY	ジョブが失敗し、再試行のためにキューに入れられた。
10	SENDING_OUTPUT	ジョブが完了し、出力を返している。

SELECT アクセス権限が付与されているユーザーは、この表を表示できます。これによって、関心のあるジョブ送信を問い合せて、ジョブの現行ステータスを確認できます。Oracle9iAS Reports Server Queue 画面を実装することで、ユーザーはこのデータのビューも表示できるようになります。このような画面を実装するには、この表に直接基づいたレポートを作成します。それによって、キュー・レポートがユーザーによるジョブ送信として表示されます。

逆に言えば、ジョブ送信のステータスで表をリアルタイムに更新することで、管理者は同時に何人のユーザーが Reports Server でジョブを実行するように要求したかを正確かつきわめて簡単に確認できます。

RW_SERVER_QUEUE 表で、ジョブがキューに入っていてまだ完了していないことをステータス・コードが示しているエントリの数をカウントすることで、サーバー上で現在アクティブなユーザーの正確な数がわかります。たとえば、次の問合せを使用できます。

```
SELECT Count (*)
FROM   RW_SERVER_QUEUE
WHERE  STATUS_CODE IN (1,      -- ENQUEUED
                       2,      -- OPENING
                       3)      -- RUNNING
AND    JOB_TYPE != 'Scheduled'
```

注意： 表には、レポートがキューに入れられ、実行され、完了したときの日付と時刻が含まれています。ただし、ジョブには QUEUED および STARTED の時間が定義されていて FINISHED の値がないという事実に基づいて問合せを使用することはお薦めできません。無効な入力などの予期しないエラーが発生してレポートが終了した場合は、FINISHED 列が NULL のままとなります。ただし STATUS_CODE と STATUS_MESSAGE は、どちらもエラーが発生したこと、およびそのエラーの原因を示します。

14.5.4 キュー・アクティビティによるデータベースの更新

Reports Server のジョブ・キューは、PL/SQL ベース API を使用することで実装されます。この API には、要求が行われたときにキュー情報でキュー・テーブルを更新する機能があります。この実装は、次のパスに定義されます。

```
%ORACLE_HOME%\reports\admin\sql\rw_server.sql
```

キューを実装する手順は、次のとおりです。

1. `rw_server.sql` ファイルをデータベースにロードします（このファイルは、Oracle9iAS Reports Services のインストール先である `%ORACLE_HOME%\reports\admin\sql` に含まれています）。

これによって、レポート・キューの情報を所有し、サーバー・キュー API に対する実行特権を持つスキーマが作成されます。
2. Oracle9iAS Reports Server 構成ファイル（`%ORACLE_HOME%\reports\conf\server_name.conf`）に含まれる `jobStatusRepository` 要素の `REPOSITORYCONN` 属性を、キュー・データを所有するスキーマの接続文字列に設定します。

サーバーは、起動時に定義済みのユーザーとして接続し、ジョブ送信をログに記録します。

注意： Oracle9iAS Reports Server および Oracle データベースが 1 台のスタンドアロン Windows NT マシンにインストールされている場合に、`REPOSITORYCONN` を定義すると、Windows NT のブート時に Oracle9iAS Reports Services が自動的に起動されなくなる可能性があります。この場合、Oracle データベース・サービスが開始されていない可能性があるため、Oracle9iAS Reports Server が要求されたログインを実行できません。Oracle データベースが開始されれば、Oracle9iAS Reports Server を手動で起動できます。

第 V 部

付録

第 V 部は、Oracle9iAS Reports Services 環境での動作に関する追加の詳細情報が記載されている付録です。これには、**Reports** コマンドやそれに関連するコマンドライン引数の情報だけでなく、**Reports** 関連の環境変数に関する詳細も記載されています。

第 V 部には、次の付録があります。

- [付録 A 「コマンドライン引数」](#)
- [付録 B 「Reports 関連の環境変数」](#)
- [付録 C 「Oracle9iAS Portal へのレポートのバッチ登録」](#)

コマンドライン引数

この付録では、`rwclient`、`rwrn`、`rwbuilder`、`rwconverter`、`rwervlet`、`rwcgi`、`rwserver` などのコマンドとともに使用できるコマンドライン引数について説明し、例を示します。各引数の説明には、どのコマンドでどの引数キーワードを使用できるかを示した表を記載します。

注意： ランタイム URL でのコマンドライン引数の使用例は、[第 8 章「レポート要求の実行」](#)を参照してください。

この付録には、次の項があります。

- [コマンドの概要](#)
- [コマンドライン構文](#)
- [全般的な使用上の注意](#)
- [コマンドライン引数](#)

A.1 コマンドの概要

この項では、この付録で説明するキーワードと引数を使用するコマンドについて簡単に説明します。これには、次のようなコマンドがあります。

- [rwclient](#)
- [rwrn](#)
- [rwbuilder](#)
- [rwconverter](#)
- [rwservlet](#)
- [rwcgi](#)
- [rwserver](#)

各コマンドの説明では、使用できるキーワードのリストを記載します。キーワードは、引数とともにコマンドラインに入力する必要があります。コマンドラインに入力する際、キーワードは順不同で使用できます。

A.1.1 rwclient

`rwclient` コマンドは、コマンドラインを解析し、指定された（またはデフォルトの）Reports Server に転送します。

rwclient とともに使用するキーワード

このリストでキーワードを囲む括弧は、キーワード間を区切る目的のみで挿入されています。

```
rwclient [ACCESSIBLE] [ARRAYSIZE] [AUTHID] [AUTOCOMMIT] [BCC] [BLANKPAGES] [BUFFERS]
[CACHELOB] [CC] [CELLWRAPPER] [CMDFILE] [COPIES] [CUSTOMIZE] [DATEFORMATMASK]
[DELIMITED_HDR] [DELIMITER] [DESFORMAT] [DESNAME] [DESTINATION] [DESTYPE]
[DISTRIBUTE] [EXPIRATION] [EXPRESS_SERVER] [FROM] [IGNOREMARGIN] [JOBNAME] [JOBTYP]
[LONGCHUNK] [MODE] [MODULE|REPORT] [NONBLOCKSQL] [NOTIFYFAILURE] [ONFAILURE]
[ONSUCCESS] [ORIENTATION] [PAGESIZE] [PDFCOMP] [PDFEMBED] [READONLY] [REPLYTO]
[REPORT|MODULE] [ROLE] [RUNDEBUG] [SCHEDULE] [SERVER] [SUBJECT] [TOLERANCE]
[TRACEMODE] [TRACEOPTS] [USERID]
```

A.1.2 rwrn

rwrn コマンドは、Oracle9iAS Reports Services のインプロセス・サーバーを使用してレポートを実行します。.rep ファイルを実行する場合、PL/SQL はすでにコンパイルされているので、再コンパイルはされません。.rdf ファイルを実行する場合、PL/SQL は必要に応じて自動的に再コンパイルされます。再コンパイルが必要になるのは、レポートが Reports Builder によってコンパイルおよび保存されていない場合や、レポートを実行したプラットフォームまたはバージョンが、前回レポートがコンパイルおよび保存されたプラットフォームと互換性がない場合です。

rwrn とともに使用するキーワード

このリストでキーワードを囲む括弧は、キーワード間を区切る目的のみで挿入されています。

```
rwrn [ACCESSIBLE] [ARRAYSIZE] [AUTHID] [AUTOCOMMIT] [BCC] [BLANKPAGES] [BUFFERS]
[CACHELOB] [CC] [CELLWRAPPER] [CMDFILE] [COPIES] [CUSTOMIZE] [DATEFORMATMASK]
[DELIMITED_HDR] [DELIMITER] [DESFORMAT] [DESNAME] [DESTINATION] [DESTYPE]
[DISTRIBUTE] [EXPRESS_SERVER] [FROM] [IGNOREMARGIN] [LONGCHUNK] [MODE]
[MODULE|REPORT] [NONBLOCKSQL] [NOTIFYFAILURE] [ONFAILURE] [ONSUCCESS] [ORIENTATION]
[PAGESIZE] [PAGESTREAM] [PDFCOMP] [PDFEMBED] [PRINTJOB] [READONLY] [REPLYTO]
[REPORT|MODULE] [ROLE] [RUNDEBUG] [SAVE_RDF] [SUBJECT] [TRACEFILE] [TRACEMODE]
[TRACEOPTS] [USERID]
```

A.1.3 rwbuilder

rwbuilder コマンドは、Reports Builder を起動します。コマンド・プロンプトで rwbuilder コマンドに REPORT|MODULE キーワードを追加して [Enter] を押すと、指定されたレポートが Reports Builder ナビゲータで選択された状態で、Reports Builder が起動します。指定されたレポートがない場合、Reports Builder は、既存のレポートをオープンするか新規レポートを作成するかを選択する「ようこそ」ダイアログとともに起動します。

rwbuilder とともに使用するキーワード

このリストでキーワードを囲む括弧は、キーワード間を区切る目的のみで挿入されています。

```
rwbuilder [ACCESSIBLE] [ARRAYSIZE] [AUTOCOMMIT] [BLANKPAGES] [BUFFERS] [CACHELOB]
[CMDFILE] [EXPRESS_SERVER] [LONGCHUNK] [MODULE|REPORT] [NONBLOCKSQL] [ONFAILURE]
[ONSUCCESS] [PAGESIZE] [PRINTJOB] [READONLY] [REPORT|MODULE] [RUNDEBUG] [SAVE_RDF]
[TRACEFILE] [TRACEMODE] [TRACEOPTS] [USERID] [WEBSERVER_DEBUG] [WEBSERVER_DOCROOT]
[WEBSERVER_PORT]
```

A.1.4 rwconverter

rwconverter コマンドによって、1 つ以上のレポート定義または PL/SQL ライブラリをある保存形式から別の保存形式に変換できます。たとえば、rwconverter を使用して次のような処理ができます。

- レポート・ファイルと XML ファイルの組み合わせによる新しいレポートの作成

注意： レポートをテンプレートに変換する場合、レポートのヘッダー・セクションとトレーラ・セクションおよびマージン領域にあるオブジェクトのみがテンプレートに使用されます。メイン・セクションにあるオブジェクトは無視されます。

- .rdf ファイルに格納されたレポートを .rep、.rex または .tdf ファイルに変換
- .rex ファイルに格納されたレポートを .rdf または .tdf ファイルに変換
- データベースに格納されたライブラリを .pld または .pll ファイルに変換
- .pld ファイルに格納されたライブラリをデータベース・ライブラリまたは .pll ファイルに変換
- .pll ファイルに格納されたライブラリを .pld ファイルのデータベース・ライブラリに変換

注意： 連結ライブラリ付きのレポートを変換する場合、レポートに連結された .pll ファイルを変換してから、.rdf/.rex ファイルを変換する必要があります。

- Oracle9iAS Portal にレポートをバッチ登録する PL/SQL スクリプトのバッチ作成

rwconverter によって、変換処理の一部として、レポートの PL/SQL が自動的にコンパイルされる場合もあります。変換先が .rex ファイルでない場合、次の状況では、rwconverter によって PL/SQL が自動的にコンパイルされます。

- 変換元が .rep ファイルの場合
- ソースとして .rex ファイルを使用する場合
- ソースとして別のプラットフォーム上に作成されたレポートを使用する場合

その他の状況では、各自でレポートの PL/SQL をコンパイルする必要があります (Reports Builder の「プログラム」⇒「コンパイル」⇒「すべて」を使用します)。

注意： フォントは、変換時ではなく、Report Builder や Reports Runtime でレポートをオープンしたときにマップされます。

rwconverter とともに使用するキーワード

このリストでキーワードを囲む括弧は、キーワード間を区切る目的のみで挿入されています。

```
rwconverter [BATCH] [CMDFILE] [CUSTOMIZE] [DEST] [DTYPE] [DUNIT] [FORMSIZE]
[NOTIFYFAILURE] [OVERWRITE] [PAGESIZE] [SOURCE] [STYPE] [USERID] [P_OWNER] [P_
SERVERS] [P_NAME] [P_DESCRIPTION] [P_PRIVILEGE] [P_AVAILABILITY] [P_TYPES] [P_
FORMATS] [P_PRINTERS] [P_PFORMTEMPLATE] [P_TRIGGER]
```

A.1.5 rwservlet

rwservlet コマンドは、HTTP Server と Reports Server 間で情報を変換および配布します。

注意： rwservlet コマンドを使用して JSP を実行する場合、rwservlet のキーワードをすべて使用できます。rwservlet による JSP 実行の詳細は、[第 8 章「レポート要求の実行」](#)を参照してください。

rwservlet とともに使用するキーワード

このリストでキーワードを囲む括弧は、キーワード間を区切る目的のみで挿入されています。

```
rwservlet [ACCESSIBLE] [ARRAYSIZE] [AUTHID] [AUTOCOMMIT] [BCC] [BLANKPAGES]
[BUFFERS] [CACHELOB] [CC] [CELLWRAPPER] [CMDKEY] [CONTENTAREA] [COPIES] [CUSTOMIZE]
[DATEFORMATMASK] [DELAUTH] [DELIMITED_HDR] [DELIMITER] [DESFORMAT] [DESNAME]
[DESTINATION] [DESTYPE] [DISTRIBUTE] [EXPIRATION] [EXPIREDAYS] [EXPRESS_SERVER]
[FROM] [GETJOBID] [GETSERVERINFO] [HELP] [IGNOREMARGIN] [ITEMTITLE] [JOBNAME]
[JOBTYP] [KILLJOBID] [LONGCHUNK] [MODE] [MODULE|REPORT] [NONBLOCKSQL]
[NOTIFYFAILURE] [NOTIFYSUCCESS] [NOTIFYFAILURE] [ONFAILURE] [ONSUCCESS]
[ORIENTATION] [OUTPUTFOLDER] [OUTPUTPAGE] [PAGEGROUP] [PAGESIZE] [PAGESTREAM]
[PARAMFORM] [PARSEQUERY] [PDFCOMP] [PDFEMBED] [READONLY] [REPLACEITEM] [REPLYTO]
[REPORT|MODULE] [ROLE] [RUNDEB] [SCHEDULE] [SERVER] [SHOWENV] [SHOWENV] [SHOWMAP]
[SHOWJOBS] [SHOWMYJOBS] [SITENAME] [STATUSFORMAT] [STATUSFOLDER] [STATUSPAGE]
[SSOCONN] [SUBJECT] [TOLERANCE] [TRACEMODE] [TRACEOPTS] [URLPARAMETER] [USERID]
```

注意： キーワード `showenv`、`showjobs`、`showmap`、`showmyjobs`、`showjobid`、`killjobid`、`parsequery`、`showauth`、`delauth`、`getjobid` および `getserverinfo` は、キーワードと値のペアではなくコマンドです。つまり、対応する値を持たず、コマンドだけが入力されます。これらのキーワードの詳細は、構文説明を参照してください。

A.1.6 rwcgi

`rwserverlet` コマンドと同じく、`rwcgi` コマンドは、HTTP Server と Reports Server 間で情報を変換および配布します。`rwcgi` よりも `rwserverlet` コマンドを使用するようお勧めします。Reports CGI は、下位互換性を確保するためにのみ維持されています。

rwcgi とともに使用するキーワード

このリストでキーワードを囲む括弧は、キーワード間を区切る目的のみで挿入されています。

`rwcgi` [ACCESSIBLE] [ARRAYSIZE] [AUTHID] [AUTOCOMMIT] [BCC] [BLANKPAGES] [BUFFERS] [CACHELOB] [CC] [CELLWRAPPER] [CONTENTAREA] [COPIES] [CUSTOMIZE] [DATEFORMATMASK] [DELIMITED_HDR] [DELIMITER] [DESFORMAT] [DESNAME] [DESTINATION] [DESTYPE] [DISTRIBUTE] [EXPIRATION] [EXPIREDAYS] [EXPRESS_SERVER] [FROM] [IGNOREMARGIN] [ITEMTITLE] [JOBNAME] [JOBTYP] [LONGCHUNK] [MODE] [MODULE|REPORT] [NONBLOCKSQL] [NOTIFYFAILURE] [NOTIFYSUCCESS] [NOTIFYFAILURE] [ONFAILURE] [ONSUCCESS] [ORIENTATION] [OUTPUTFOLDER] [OUTPUTPAGE] [PAGEGROUP] [PAGESIZE] [PAGESTREAM] [PARAMFORM] [PDFCOMP] [PDFEMBED] [READONLY] [REPLACEITEM] [REPLYTO] [REPORT|MODULE] [ROLE] [RUNDEB] [SCHEDULE] [SERVER] [SITENAME] [STATUSFOLDER] [STATUSPAGE] [SUBJECT] [TOLERANCE] [TRACEMODE] [TRACEOPTS] [USERID]

A.1.7 rwserver

`rwserver` コマンドは、Reports Server を起動します。Reports Server は、クライアント要求を処理します。この処理では、クライアント要求を、認証と許可の確認、スケジューリング、キャッシュ、および配布（カスタムまたは Pluggable Destinations への配布など）など、各種サービスに割り当てます。また、要求されたレポートを生成するランタイム・エンジンを起動したり、生成されたレポートを Reports キャッシュから取り出したり、ジョブの準備が完了したことをクライアントに通知したりします。

rwserver とともに使用するキーワード

このリストでキーワードを囲む括弧は、キーワード間を区切る目的のみで挿入されています。

[AUTHID] [AUTOSTART] [BATCH] [SERVER] [SHUTDOWN] [TRACEOPTS] [INSTALL] [UNINSTALL]

A.2 コマンドライン構文

コマンドラインの構文は、次のとおりです。keyword=value は、有効なコマンドライン引数です。

```
rwclient REPORT|MODULE=runfile USERID=username/password@database
[ [keyword=]value|(value1, value2, ...) ] SERVER=server_name
```

キーワードは必ず指定する必要がありますが、コマンドの後ろに順不同で使用できます。

A.3 全般的な使用上の注意

- クライアントのコマンドラインで指定されたすべてのファイル名とパスは、サーバー・マシン上のファイルとディレクトリを参照します。ただし、コマンド・ファイル・キーワード (CMDFILE=) に対して指定されたファイルはこれらを参照しません。この場合は、指定された CMDFILE が読み込まれ、元のコマンドライン (CMDFILE が記述されているコマンドライン) に追加されてから、Reports Server に送信されます。ランタイム・エンジンによって CMDFILE が再び読み込まれることはありません。
- キーワード値にファイル名を含むパスを指定しないと、Reports Server は REPORTS_PATH 環境変数からファイル検索を試みます。
- コマンドラインに CMDFILE= が含まれている場合、コマンド・ファイルが元のコマンドラインに読み込まれて追加されてから、Oracle9iAS Reports Server に送信されます。コマンド・ファイルは、Oracle9iAS Reports Engine によって再度読み込まれることはありません (「CMDFILE」を参照してください)。

A.4 コマンドライン引数

A.4.1 ACCESSIBLE

表 A-1 では、ACCESSIBLE キーワードを使用できるコマンドを示します。

表 A-1 ACCESSIBLE キーワードを使用できるコマンド

rwclient	rwrund	rwbuilder	rwconverter	rwervlet	rwcgi	rwserver
○	○	○	×	○	○	×

説明 ACCESSIBLE を使用して、Reports のアクセシビリティ関連の機能が PDF 出力で有効化されるか (YES) または無効化されるか (NO) を指定します。NO は無効化されることを示します。

構文

ACCESSIBLE={YES|NO}

値 YES は、アクセシビリティの機能がレポートの PDF 出力で有効化されることを示します。

デフォルト NO

A.4.2 ARRAYSIZE

表 A-2 では、ARRAYSIZE キーワードを使用できるコマンドを示します。

表 A-2 ARRAYSIZE キーワードを使用できるコマンド

rwclient	rwrn	rwbuilder	rwconverter	rwservlet	rwcgi	rwserver
○	○	○	×	○	○	×

説明 ARRAYSIZE を使用して、ORACLE 配列処理で使用するサイズ (KB) を指定します。通常、配列サイズが大きくなるに従って、レポートの実行速度も高まります。

構文 ARRAYSIZE=*n*

値 1 ～ 9999 までの数字を使用します (位取りのカンマは使用しないでください)。Oracle9iAS Reports Runtime では、レポートでの問合せごとに、指定された KB 数のメモリーを使用できます。

デフォルト 10

使用上の注意 ARRAYSIZE は、JSP として実行されるジョブでも使用できます。

A.4.3 AUTHID

表 A-3 では、AUTHID キーワードを使用できるコマンドを示します。

表 A-3 AUTHID キーワードを使用できるコマンド

rwclient	rwrn	rwbuilder	rwconverter	rwservlet	rwcgi	rwserver
○	○	×	×	○	○	○

説明 AUTHID を使用して、制限付きの Oracle9iAS Reports Server のユーザーを認証するのに使用されるユーザー名、およびオプションでパスワードを指定します。ユーザー認証によって、レポート要求を行っているユーザーが要求されたレポートを実行するアクセス権限を持っていることが保証されます。

構文 AUTHID=username[/password]

値 Oracle9iAS Portal で生成した有効なユーザー名、およびオプションでパスワードを指定します。Oracle9iAS Portal に新しいユーザー・アカウントを生成するときは、データベース管理者に問い合わせてください。

デフォルト なし

使用上の注意 AUTHID は、JSP として実行されるジョブでも使用できます。

シングル・サインオン環境では、Oracle Single Sign-on Server が認証処理を実行し、AUTHID 内のユーザー名のみが Reports Server に渡されます。

A.4.4 AUTOCOMMIT

表 A-4 では、AUTOCOMMIT キーワードを使用できるコマンドを示します。

表 A-4 AUTOCOMMIT キーワードを使用できるコマンド

rwclient	rwrn	rwbuilder	rwconverter	rwervlet	rwcgi	rwserver
○	○	○	×	○	○	×

説明 AUTOCOMMIT を使用して、データベースの変更（CREATE など）をデータベースに自動的にコミットするかどうかを指定します。ある種の非 Oracle データベース（SQL Server など）では、AUTOCOMMIT=YES とする必要があります。

構文 AUTOCOMMIT={YES|NO}

値 YES または NO

デフォルト NO

使用上の注意 AUTOCOMMIT は、JSP として実行されるジョブでも使用できます。

A.4.5 AUTOSTART

表 A-5 では、AUTOSTART キーワードを使用できるコマンドを示します。

表 A-5 AUTOSTART キーワードを使用できるコマンド

rwclient	rwrn	rwbuilder	rwconverter	rwervlet	rwcgi	rwserver
×	×	×	×	×	×	○

説明 AUTOSTART を使用して、Reports Server が初回インストール後にリポートした後、ユーザー・ログインなしで自動的に起動することを指定します。

構文 AUTOSTART={YES|NO}

値 YES または NO

デフォルト NO

使用上の注意 AUTOSTART キーワードは、Microsoft Windows プラットフォームのみで認識されます。

A.4.6 BATCH

表 A-6 では、BATCH キーワードを使用できるコマンドを示します。

表 A-6 BATCH キーワードを使用できるコマンド

rwclient	rwrn	rwbuilder	rwconverter	rwervlet	rwcgi	rwserver
×	×	×	○	×	×	○

説明 rwconverter で BATCH を使用すると、レポートまたはライブラリの変換にユーザが介入しないように、すべての端末入力および出力を非表示にします。rwserver で BATCH を使用すると、サーバー・ウィンドウ・ダイアログがオフ (YES) またはオン (NO) になり、プロセスのメッセージが表示または非表示になります。

すべての関連コマンドで、BATCH オプションは、非 UI モードでの実行をサーバーに指示します。コマンドでの使用方法は、アプリケーションによって UI が生成されないという点で、BATCH=YES を含むコマンドラインからの実行と同じです。たとえば、rwserver でこのオプションを使用すると、サーバーをスクリプトおよびリモート・エージェントから実行することができ、実行中にサーバー・ダイアログが表示されません。

構文 BATCH={YES|NO}

デフォルト NO

A.4.7 BCC

表 A-7 では、BCC キーワードを使用できるコマンドを示します。

表 A-7 BCC キーワードを使用できるコマンド

rwclient	rwrn	rwbuilder	rwconverter	rwervlet	rwcgi	rwserver
○	○	×	×	○	○	×

説明 BCC を使用して、電子メールの BCC 宛先を指定します。

注意： BCC 宛先とは、指定された受信者名が他の受信者に見えない（公開されない）宛先をいいます。

構文 BCC=someone@foo.com OR BCC="someone@foo.com,sometwo@foo.com"

値 1つ以上の有効な電子メール・アドレス

デフォルト なし

使用上の注意 複数の電子メール・アドレスを指定するには、アドレスのリストを引用符で囲み、リスト内のアドレスをカンマで区切ります。

関連するキーワードは、[BCC](#)、[CC](#)、[FROM](#)、[REPLYTO](#) および [SUBJECT](#) です。電子メールの主な宛先を指定するには、[DESNAME](#) を使用します。

BCC は、JSP として実行されるジョブでも使用できます。

A.4.8 BLANKPAGES

[表 A-8](#) では、BLANKPAGES キーワードを使用できるコマンドを示します。

表 A-8 BLANKPAGES キーワードを使用できるコマンド

rwclient	rwrn	rwbuilder	rwconverter	rwservlet	rwsgi	rwserver
○	○	○	×	○	○	×

説明 BLANKPAGES を使用して、レポートの印刷時に空白ページを抑止するかどうか指定します。このキーワードは、レポート出力に印刷したくない空白ページがある場合に使用します。

構文 BLANKPAGES={YES|NO}

値 YES はすべての空白ページを印刷します。NO は空白ページを印刷しません。

デフォルト YES

使用上の注意 BLANKPAGES は、論理ページが複数の物理ページ（またはパネル）にわたっていて、空白の物理ページの印刷はすべて抑止するときに特に便利です。

A.4.9 BUFFERS

表 A-9 では、BUFFERS キーワードを使用できるコマンドを示します。

表 A-9 BUFFERS キーワードを使用できるコマンド

rwclient	rwrund	rwbuilder	rwconverter	rwervlet	rwcgi	rwserver
○	○	○	×	○	○	×

説明 BUFFERS を使用して、仮想メモリー・キャッシュのサイズを KB で指定します。レポートの実行に十分な領域を確保する一方で、システムのリソースを使い過ぎないように調整する必要があります。

構文 BUFFERS=*n*

値 1 ～ 9999 までの数字を使用します（位取りのカンマや小数点などは使用しないでください）。オペレーティング・システムによっては、上限がこれより低い場合もあります。

デフォルト 640

使用上の注意 この設定がセッションの途中で変更されても、変更は次のレポートが実行されるまで有効になりません。

BUFFERS は、JSP として実行されるジョブでも使用できます。

A.4.10 CACHELOB

表 A-10 では、CACHELOB キーワードを使用できるコマンドを示します。

表 A-10 CACHELOB キーワードを使用できるコマンド

rwclient	rwrund	rwbuilder	rwconverter	rwervlet	rwcgi	rwserver
○	○	○	×	○	○	×

説明 CACHELOB を使用して、取り出した Oracle8 のラージ・オブジェクトを Reports Server の一時ファイル・ディレクトリにキャッシュするかどうかを指定します。一時ファイル・ディレクトリは、環境変数 REPORTS_TMP、または *tempDir* プロパティ（Reports Server 構成ファイル <*server_name*>.conf> の *engine* 要素のプロパティ）で指定されます。*tempDir* 設定は REPORTS_TMP 設定よりも優先されます。

構文 CACHELOB=NO

値 YES の場合、LOB を一時ファイル・ディレクトリにキャッシュします。NO の場合、LOB を一時ファイル・ディレクトリにキャッシュしません。

デフォルト YES

使用上の注意

- このオプションは、コマンドラインでのみ設定できます。
- サーバー上の一時ファイル・ディレクトリの位置に十分なディスク領域がない場合、この値は NO に設定することをお薦めします。ただし、値を NO に設定すると、LOB をデータベースから複数回フェッチする必要が生じるため、パフォーマンスが低下する可能性があります。
- CACHELOB は、JSP として実行されるジョブでも使用できます。

A.4.11 CC

表 A-11 では、CC キーワードを使用できるコマンドを示します。

表 A-11 CC キーワードを使用できるコマンド

rwclient	rwrn	rwbuilder	rwconverter	rwservlet	rwcgi	rwserver
○	○	×	×	○	○	×

説明 CC を使用して、電子メールの CC 宛先を指定します。

構文 CC=someone@foo.com OR CC="someone@foo.com,sometwo@foo.com"

値 1 つ以上の有効な電子メール・アドレス

デフォルト なし

使用上の注意 複数の電子メール・アドレスを指定するには、アドレスのリストを引用符で囲み、リスト内のアドレスをカンマで区切ります。

関連するキーワードは、BCC、CC、FROM、REPLYTO および SUBJECT です。電子メールの主な宛先を指定するには、DESNAME を使用します。

A.4.12 CELLWRAPPER

表 A-12 では、CELLWRAPPER キーワードを使用できるコマンドを示します。

表 A-12 CELLWRAPPER キーワードを使用できるコマンド

rwclient	rwrn	rwbuilder	rwconverter	rwservlet	rwcgi	rwserver
○	○	×	×	○	○	×

説明 CELLWRAPPER を使用して、デリミタ付きレポート出力においてセルの前後に表示される文字を指定します。

構文 CELLWRAPPER=value

値 任意の英数字または英数字の文字列を使用します。

" セルの両側に二重引用符が配置されます。

' セルの両側に一重引用符が配置されます。

次の 4 つの予約済みの値を使用することもできます。

tab セルの両側にタブが配置されます。

space セルの両側にシングル・スペースが配置されます。

return セルの両側に改行が配置されます。

none セル・ラッパーは使用されません。

ASCII キャラクタ・セットに基づくエスケープ・シーケンスを使用することもできます。

¥t セルの両側にタブが配置されます。

¥n セルの両側に改行が配置されます。

デフォルト なし

使用上の注意

- このキーワードを使用できるのは、DESFORMAT=DELIMITED と指定しているときのみです。
- セル・ラッパーは実際のデリミタとは異なります。セル・ラッパーは、デリミタ付きデータの周囲に表示される文字を指定します。デリミタは、2 つのデータ間の境界または区切り位置を示します。

A.4.13 CMDFILE

表 A-13 では、CMDFILE キーワードを使用できるコマンドを示します。

表 A-13 CMDFILE キーワードを使用できるコマンド

rwclient	rwrund	rwbuilder	rwconverter	rwservlet	rwcgi	rwserver
○	○	○	○	×	×	×

説明 CMDFILE を使用して、レポートのコマンドライン引数を格納するファイルと呼び出します。呼び出すファイルは、.txt またはその他の ASCII 形式ファイルである必要があります。

CMDFILE と cgicmd.dat ファイルの違いは、CMDFILE には 1 つのレポートに対する 1 つのコマンドラインを指定できるのに対し、cgicmd.dat ファイルには、複数のレポートに対して、キーによって識別される複数のコマンドを格納できる点です。

CMDFILE キーワードを使用すると、実行コマンドを呼び出すたびに多くの引数を指定しなくても、レポートを実行できます。

構文 CMDFILE=cmdfile

値 任意の有効なコマンド・ファイル

デフォルト なし

使用上の注意

- rwservlet および rwcgi では、CMDFILE キーワードを使用するのではなく、cgicmd.dat ファイルのキーを参照する *key* 引数を使用します。
- コマンド・ファイルは、別のコマンド・ファイルを参照できます。
- コマンド・ファイルで指定するコマンドライン構文は、コマンドラインで使用される構文と同じです。
- コマンドラインに入力した値は、コマンド・ファイルで指定した値を上書きします。たとえば、コマンドラインから rwclient を COPIES=1 および CMDFILE=RUNONE (コマンド・ファイル) として指定するとします。一方 RUNONE ファイルには COPIES の値として 2 が指定されています。コマンドラインで指定した COPIES の値 (1) は、RUNONE ファイルで指定した COPIES の値 (2) を上書きします。この場合、生成されるレポートのコピーは 1 つのみです。
- このキーワードの値は、オペレーティング・システムによって異なります。

A.4.14 CMDKEY

表 A-14 では、CMDKEY キーワードを使用できるコマンドを示します。

表 A-14 CMDKEY キーワードを使用できるコマンド

rwclient	rwrund	rwbuilder	rwconverter	rwservlet	rwcgi	rwserver
×	×	×	×	○	×	×

説明 CMDKEY を使用して、cgicmd.dat ファイル内の、キーによって識別されるコマンドラインを呼び出します。次に例を示します。

http:// .../reports/rwervlet?cmddkey=key& ...

注意： CMDKEY は、JSP として実行されるモジュールでも使用できます。詳細は、[第 8 章「レポート要求の実行」](#)を参照してください。

構文 CMDKEY=key

値 cgicmd.dat ファイルで指定されたコマンドラインに関連付けられたキーの名前

デフォルト なし

使用上の注意 CMDKEY を rwervlet で使用する際、コマンドラインでは順不同で使用できます（URL の場合は、疑問符の後に使用します）。rwervlet には、CMDKEY とともに他のコマンドライン・キーワードも使用できます。

CMDKEY は、JSP として実行されるジョブでも使用できます。

A.4.15 CONTENTAREA

[表 A-15](#) では、CONTENTAREA キーワードを使用できるコマンドを示します。

表 A-15 CONTENTAREA キーワードを使用できるコマンド

rwclient	rwrn	rwbuilder	rwconverter	rwervlet	rwcgi	rwserver
○	○	×	×	○	○	×

説明 CONTENTAREA を使用して、レポート出力の配信先となる Oracle Portal 内のコンテンツ領域を指定します。このキーワードは、Oracle Portal の旧バージョン（3.0.9）に対する下位互換性を確保するために維持されています。Oracle9iAS Portal では、[PAGEGROUP](#) を使用してください（[「SITENAME」](#)も参照してください）。

構文 CONTENTAREA="Name of Portal content area"

値 有効な Oracle Portal コンテンツ領域の名前

デフォルト なし

使用上の注意 レポート出力を Oracle Portal に配信する際、このキーワードの使用は必須です。値に文字スペース単位が含まれる場合や、cgicmd.dat ファイルで引数を指定する場合は、値の前後に引用符を付けます。

関連するキーワードは、[CONTENTAREA](#)、[EXPIREDAYS](#)、[ITEMTITLE](#)、[OUTPUTFOLDER](#)、[OUTPUTPAGE](#)、[PAGEGROUP](#)、[REPLACEITEM](#)、[SCHEDULE](#)、[SITENAME](#)、[STATUSFOLDER](#) および [STATUSPAGE](#) です。

ページやページ・グループなどの Oracle Portal オブジェクトには、表示名と内部名という 2 つの名前があります。Oracle Portal 出力で使用するオブジェクトを Oracle Portal 内で作成する際、Oracle Portal のオンライン・ヘルプで指定された内部名の命名規則に従って、内部名と表示名を同じにします。これにより、Reports コマンドラインで Portal 関連のキーワードに値を指定する際、どちらの名前を指定すればよいか考慮する必要がなくなります。

A.4.16 COPIES

表 A-16 では、COPIES キーワードを使用できるコマンドを示します。

表 A-16 COPIES キーワードを使用できるコマンド

rwclient	rwrn	rwbuilder	rwconverter	rwervlet	rwcgi	rwserver
○	○	×	×	○	○	×

説明 COPIES を使用して、印刷するレポート出力の部数を指定します。

構文 COPIES=*n*

値 1 ～ 9999 までの整数を使用します（位取りのカンマや小数点などは使用しないでください）。

デフォルト COPIES パラメータの Initial Value プロパティから取得します (Initial Value は、設計時に Reports Builder で定義されています)。

使用上の注意

- DESTYPE が PRINTER でない場合、このキーワードは無視されます。
- ランタイム・パラメータ・フォーム上で COPIES が空白のままになっている場合は、デフォルトの 1 が使用されます。

A.4.17 CUSTOMIZE

表 A-17 では、CUSTOMIZE キーワードを使用できるコマンドを示します。

表 A-17 CUSTOMIZE キーワードを使用できるコマンド

rwclient	rwrn	rwbuilder	rwconverter	rwservlet	rwcgi	rwserver
○	○	×	○	○	○	×

説明 CUSTOMIZE を使用して、現在のレポートに対して Reports XML ファイルが実行されるよう指定します。Reports XML ファイルには、レポート定義をなんらかの形で変更する、カスタマイズされた設定（レイアウトやデータ・モデルの変更など）が入っています。

構文 CUSTOMIZE=filename.xml | (filename1.xml, filename2.xml, ...)

値 有効な XML レポート定義を含むファイル名またはファイル名のリストを指定します。必要に応じて、ファイル名の前にパス情報を付加します（REPORTS_PATH レジストリまたは engine 要素の SourceDir プロパティで指定されているパスにファイルが置かれていない場合は、パスを付加する必要があります）。

注意： XML カスタマイズ・ファイルを使用してレポートを実行時にカスタマイズする方法の詳細は、第 10 章「XML によるレポートのカスタマイズ」を参照してください。

デフォルト なし

使用上の注意 CUSTOMIZE は、JSP として実行されるジョブでも使用できます。

A.4.18 DATEFORMATMASK

表 A-18 では、DATEFORMATMASK キーワードを使用できるコマンドを示します。

表 A-18 DATEFORMATMASK キーワードを使用できるコマンド

rwclient	rwrn	rwbuilder	rwconverter	rwservlet	rwcgi	rwserver
○	○	×	×	○	○	×

説明 DATEFORMATMASK を使用して、日付の値がデリミタ付きのレポート出力でどのように表示されるかを指定します。

構文 DATEFORMATMASK=mask

値 任意の日付書式マスク

デフォルト なし

使用上の注意 このキーワードを使用できるのは、DESFORMAT=DELIMITED と指定しているときのみです。

注意： 有効な DATEFORMATMASK 値は、Report Builder のオンライン・ヘルプのトピック「DATE および TIME 書式マスク構文」を参照してください。

DATEFORMATMASK は、JSP として実行されるジョブでも使用できます。

A.4.19 DELAUTH

表 A-19 では、DELAUTH キーワードを使用できるコマンドを示します。

表 A-19 DELAUTH キーワードを使用できるコマンド

rwclient	rwrwn	rwbuilder	rwconverter	rwervlet	rwsgi	rwserver
×	×	×	×	○	○	×

説明 DELAUTH を使用して、rwervlet または rwsgi のユーザー ID Cookie を削除します。

構文 `http://yourwebserver/yourervletpath/rwervlet/DELAUTH[?]
[server=servername] [&authid=username/password]`

値 構文を参照してください。

デフォルト なし

使用上の注意 関連するキーワードは、[SERVER](#) および [AUTHID](#) です。

A.4.20 DELIMITED_HDR

表 A-20 では、DELIMITED_HDR キーワードを使用できるコマンドを示します。

表 A-20 DELIMITED_HDR キーワードを使用できるコマンド

rwclient	rwrn	rwbuilder	rwconverter	rwervlet	rwcgi	rwserver
○	○	×	×	○	○	×

説明 DELIMITED_HDR を使用して、DESFORMAT=DELIMITED によるレポート実行の際に、ヘッダーなどのボイラープレートを手をすべてオフにします。

構文 DELIMITED_HDR={YES|NO}

値 YES は、デリミタ付き出力ファイルのボイラープレート・テキストを手をすべてオフにします。NO は、デリミタ付き出力ファイルのボイラープレート・テキストをそのまま残します。

デフォルト YES

使用上の注意 このキーワードを使用できるのは、DESFORMAT=DELIMITED と指定しているときのみです。

A.4.21 DELIMITER

表 A-21 では、DELIMITER キーワードを使用できるコマンドを示します。

表 A-21 DELIMITER キーワードを使用できるコマンド

rwclient	rwrn	rwbuilder	rwconverter	rwervlet	rwcgi	rwserver
○	○	×	×	○	○	×

説明 DELIMITER を使用して、レポート出力のセルを区切るために使用する文字を指定します。

構文 DELIMITER=value

値 次のような任意の英数字または英数字の文字列を使用します。

- 、 各セルをカンマで区切ります。
- 。 各セルをピリオドで区切ります。

次の 4 つの予約済みの値を使用することもできます。

- tab 各セルをタブで区切ります。
- space 各セルをスペースで区切ります。
- return 各セルを改行で区切ります。
- none デリミタを使用しません。

ASCII キャラクタ・セットに基づくエスケープ・シーケンスを使用することもできます。

- ¥t 各セルをタブで区切ります。
- ¥n 各セルを改行で区切ります。

デフォルト Tab

使用上の注意 このキーワードを使用できるのは、**DESFORMAT=DELIMITED** と指定しているときのみです。

A.4.22 DESFORMAT

表 A-22 では、DESFORMAT キーワードを使用できるコマンドを示します。

表 A-22 DESFORMAT キーワードを使用できるコマンド

rwclient	rwrund	rwbuilder	rwconverter	rwservlet	rwcgi	rwserver
○	○	×	×	○	○	×

説明 ジョブ出力のフォーマットを指定します。ビットマップ環境では、DESFORMAT を使用して、DESTYPE が FILE であるときに使用されるプリンタ・ドライバを指定します。キャラクタ・モード環境では、DESNAME で指定されたプリンタの特徴を指定します。

構文 DESFORMAT=*desformat*

値 長さが 1KB を超えない任意の有効な宛先フォーマットです。このキーワードの有効値の例は、表 A-23 を参照してください。

表 A-23 DESFORMAT の有効値

値	説明
CHARACTER	MODE が CHARACTER のとき、DESFORMAT は、hpl、hplwide、dec、decwide、decland、dec180、df1t または wide などのプリンタ定義を指定します。

表 A-23 DESFORMAT の有効値（続き）

値	説明
DELIMITED	このレポート出力は、標準のスプレッドシート・ユーティリティ (Microsoft Excel など) で読み込むことのできるファイルに出力されます。デリミタを選んでいない場合、デフォルトのデリミタとして TAB が使用されます。
HTML	このレポート出力は、HTML 形式のファイルに出力されます。
HTMLCSS	このレポート出力は、スタイル・シート拡張を含むファイルに出力されません。
PDF	このレポート出力は、PDF 形式のファイルに出力され、Adobe Acrobat などの PDF ビューアで読み込むことができます。
POSTSCRIPT	このレポート出力は、ポストスクリプト形式のファイルに出力されます。
RTF	RTF 形式。このレポート出力は、ワード・プロセッサ (Microsoft Word など) で読み込むことのできるファイルに出力されます。MS Word でファイルをオープンする場合、レポート内のすべての図形およびオブジェクトを表示するには、「表示」⇒「ページ・レイアウト」を選択する必要があります。
XML	このレポート出力は、XML ファイルとして保存されます。このレポートは、XML をサポートしているブラウザ、あるいは任意の XML 表示アプリケーションでオープンしたり読み込んだりできます。

デフォルト DESFORMAT パラメータの Initial Value プロパティから取得します (Initial Value は、設計時に Reports Builder で定義されています)。Report Builder でレポートを実行する際に DESFORMAT が空白または df1t である場合、現行のプリンタ・ドライバ (「ファイル」⇒「プリンタの選択」で指定) が使用されます。「プリンタの選択」で何も選択していない場合、デフォルトで PostScript が使用されます。

使用上の注意 このキーワードの値は、オペレーティング・システムによっては、大文字と小文字の区別が必要な場合があります。

A.4.23 DESNAME

表 A-24 では、DESNAME キーワードを使用できるコマンドを示します。

表 A-24 DESNAME キーワードを使用できるコマンド

rwclient	rwrn	rwbuilder	rwconverter	rwservlet	rwcgi	rwserver
○	○	×	×	○	○	×

説明 DESNAME を使用して、レポート出力の送信先のキャッシュ、ファイル、プリンタ、Oracle9iAS Portal、または電子メール ID（または配布リスト）を指定します。レポート出力を電子メールで送信するには、電子メール ID を通常の電子メール・アプリケーション（任意の SMTP 互換アプリケーション）の場合と同様に指定します。複数のユーザー名を指定する際は、スペースを入れずにカンマで区切ります。次に例を示します。

```
name,name,name
```

構文 DESNAME=desname

値 1KB を超えない長さの任意の有効なキャッシュ宛先、ファイル名、プリンタ名、電子メール ID または OraclePortal を指定します。プリンタ名には、オプションでポートを指定できます。次に例を示します。

```
DESNAME=printer,LPT1:
```

```
DESNAME=printer,FILE:
```

デフォルト DESNAME パラメータの Initial Value プロパティから取得します (Initial Value は、設計時に Reports Builder で定義されています)。DESTTYPE=FILE で、かつ DESNAME が空の文字列である場合は、実行時にデフォルトの reportname.lis になります。

使用上の注意 このキーワードの引数は、使用しているオペレーティング・システムによっては、大文字と小文字の区別が必要な場合があります。

A.4.24 DEST

表 A-25 では、DEST キーワードを使用できるコマンドを示します。

表 A-25 DEST キーワードを使用できるコマンド

rwclient	rwrund	rwbuilder	rwconverter	rwervlet	rwsgi	rwserver
×	×	×	○	×	×	×

説明 DEST を使用して、変換後のレポートまたはライブラリの名前を指定します。

構文 DEST={dname | (dname1, dname2, ...) | pathname}

値 有効なレポート名、ライブラリ名またはファイル名、あるいは名前をカンマで区切り、括弧で囲んだそれらのリスト ((qanda, text, dmast) など) を指定します。

デフォルト DEST キーワードを指定しない場合、rwconverter は次のデフォルト名を使用します。

- **DTYPE** が PLDFILE のとき、DEST のデフォルト名は source.pld です。
- **DTYPE** が PLLFILE のとき、DEST のデフォルト名は source.pll です。
- **DTYPE** が RDFFILE のとき、DEST のデフォルト名は source.rdf です。
- **DTYPE** が REPFILE のとき、DEST のデフォルト名は source.rep です。
- **DTYPE** が REXFILE のとき、DEST のデフォルト名は source.rex です。
- **DTYPE** が XMLFILE のとき、DEST のデフォルト名は source.xml です。
- **DTYPE** が REGISTER のとき、DEST のデフォルト名は SQL*Plus スクリプト出力ファイルの名前 (output.sql など) です。

使用上の注意

- ファイル名のレポート / ライブラリ名のリストは、各エントリをカンマで区切り、括弧で囲む必要があります。次に例を示します。

(qanda,test,dmast) あるいは (qanda, test, dmast)
- 存在するソース名よりも指定した宛先名が多い場合、余分な宛先名は無視されます。存在するソース名よりも指定した宛先名が少ない場合、宛先名がなくなるとデフォルト名が使用されます。
- DEST キーワードの値は、オペレーティング・システムによって異なります。
- DTYPE=REGISTER のとき、複数の宛先は不要です。DEST に対して複数の SQL*Plus スクリプト・ファイル名を指定した場合、最初のエントリのみが認識され、その他は無視されます。

A.4.25 DESTINATION

表 A-26 では、DESTINATION キーワードを使用できるコマンドを示します。

表 A-26 DESTINATION キーワードを使用できるコマンド

rwclient	rwrn	rwbuilder	rwconverter	rwervlet	rwcgi	rwserver
○	○	×	×	○	○	×

説明 DESTINATION キーワードを使用すると、レポートの現行の実行処理に対応する配布を定義する XML ファイルの名前を指定できます。

構文 DESTINATION=*filename.xml*

値 レポートあるいはレポートのある部分の配布を定義する XML ファイルの名前

デフォルト なし

使用上の注意 DESTINATION キーワードを使用可能にするには、コマンドラインで **DISTRIBUTE=YES** と指定する必要があります。これら両方のキーワードを指定した場合に、**DESTYPE**、**DESNAME** および **DESFORMAT** も指定すると、これらは無視されます。

注意： 拡張配布の作成の詳細は、[第 9 章「拡張配布の作成」](#)を参照してください。

A.4.26 DESTYPE

[表 A-27](#) では、DESTYPE キーワードを使用できるコマンドを示します。

表 A-27 DESTYPE キーワードを使用できるコマンド

rwclient	rwrn	rwbuilder	rwconverter	rwervlet	rwcgi	rwserver
○	○	×	×	○	○	×

説明 DESTYPE を使用して、レポート出力を受け取るデバイス・タイプを指定できます。Reports Destination API で独自の Pluggable Destinations を作成した場合は、作成した宛先をこのようにして呼び出します。

構文 DESTYPE={cache|localFile|file|printer|sysout|mail|oraclePortal|name_of_pluggable_destination}

値 [表 A-28](#) では、DESTYPE キーワードで使用できる値を説明します。

表 A-28 DESTYPE キーワードの有効値

値	説明
cache	出力は Oracle9iAS Reports のキャッシュに直接送信されます。
localFile	rwclient、rwcgi および rwervlet に対してのみ使用できます。出力はクライアント・マシン上のファイルに同期式または非同期式で送信されます。
file	出力は DESNAME で指定したサーバー上のファイルに出力されません。
printer	出力は DESNAME で指定したサーバー上のプリンタに送信されます。Oracle9iAS Reports Server が認識できるプリンタをインストールし、稼働させている必要があります。

表 A-28 DESTYPE キーワードの有効値（続き）

値	説明
mail	出力は DESNAME で指定されたメール・ユーザーに送信されます。SMTP を使用する任意のメール・システムにメールを送信できます。
OraclePortal	出力は Oracle Portal に送信されます。関連するキーワードは、 CONTENTAREA 、 EXPIREDAYS 、 ITEMTITLE 、 OUTPUTFOLDER 、 OUTPUTPAGE 、 PAGEGROUP 、 REPLACEITEM 、 SCHEDULE 、 SITENAME 、 STATUSFOLDER および STATUSPAGE です。
sysout	rwcgi にのみ有効です。出力はクライアント・マシンのデフォルトの出力デバイスに送信され、強制的に同期呼出しになります。
<i>name_of_pluggable_destination</i>	Reports Destination API で独自の Pluggable Destinations を作成した場合は、作成した宛先をこのようにして呼び出します。

デフォルト DESTYPE パラメータの Initial Value プロパティから取得します (Initial Value は、設計時に Reports Builder で定義されています)。

A.4.27 DISTRIBUTE

表 A-29 では、DISTRIBUTE キーワードを使用できるコマンドを示します。

表 A-29 DISTRIBUTE キーワードを使用できるコマンド

rwclient	rwrn	rwbuilder	rwconverter	rwservlet	rwcgi	rwserver
○	○	×	×	○	○	×

説明 DISTRIBUTE を使用すると、レポート出力の宛先としてレポート配布定義 (設計時に Reports Builder で定義します) あるいは配布 XML ファイルで定義されている配布リスト上の複数の宛先への配布を有効化または無効化できます。

構文 DISTRIBUTE={YES|NO}

値 YES は、レポートを配布リストにある宛先に配布します。NO の場合、配布リストを無視して、[DESNAME](#)、[DESTYPE](#) および [DESFORMAT](#) パラメータの指定に従ってレポートを出力します。NO は基本的に、配布用のレポート設定を実行するデバッグ・モードで、配布を実際には実行しません。

デフォルト NO

使用上の注意 DISTRIBUTE キーワードは、[DESTINATION](#) キーワードと密接に関連して動作します。DESTINATION キーワードが効果を発揮するには、DISTRIBUTE に YES を指定する必要があります。これら両方のキーワードを指定した場合に、[DESTYPE](#)、[DESNAME](#) および [DESFORMAT](#) も指定すると、これらは無視されます。

注意： 拡張配布の作成の詳細は、[第 9 章「拡張配布の作成」](#)を参照してください。

A.4.28 DTYPE

[表 A-30](#) では、DTYPE キーワードを使用できるコマンドを示します。

表 A-30 DTYPE キーワードを使用できるコマンド

rwclient	rwrn	rwbuilder	rwconverter	rwservlet	rwcgi	rwserver
×	×	×	○	×	×	×

説明 DTYPE を使用して、レポートまたはライブラリの変換後のフォーマットを指定できます。

構文 DTYPE={PLDFILE|PLLFILE|RDFFILE|REPFIL|TDFFILE|XMLFILE|JSPFILE|REGISTER}

値 次の値が適用されます。

- PLDFILE の場合、変換された PL/SQL ライブラリは、ASCII 形式のファイルに格納されます。
- PLLFILE の場合、変換された PL/SQL ライブラリは、ソース・コードおよび P コード（コンパイル済 PL/SQL）を含むファイルに格納されます。
- RDFFILE の場合、変換されたレポートは、1 つ以上のレポート定義ファイル（.rdf 拡張子を持つファイル）に格納されます。
- REPFIL の場合、変換されたレポートは、1 つ以上のバイナリ実行ファイル（.rep 拡張子を持つファイル）に格納されます。
- REXFILE の場合、変換されたレポートは、1 つ以上のテキスト・ファイル（.rex 拡張子を持つファイル）に格納されます。
- TDFFILE の場合、レポートはテンプレート・ファイル（.tdf 拡張子を持つファイル）に変換されます。
- XMLFILE の場合、変換されたレポートは、XML ファイル（.xml 拡張子を持つファイル）に格納されます。

- JSPFILE の場合、変換されたレポートは、JSP ファイル（.jsp 拡張子を持つファイル）に格納されます。
- REGISTER の場合は、SOURCE によって指定された各レポートを RWWWVREG.REGISTER_REPORT 関数で Oracle9iAS Portal にロードするためのスクリプト・ファイルが作成されます。各ロード関数には、Oracle9iAS Portal にレポートを登録するために必要な情報が入力されます。作成された SQL*Plus のスクリプト・ファイルを Oracle9iAS Portal スキーマに対して実行すると、複数のレポートを Oracle9iAS Portal にバッチ登録できます。

デフォルト REPFILE

使用上の注意 `rwconverter` を使用して `.rep` ファイルを作成する場合、ソース・レポートの PL/SQL は、自動的にコンパイルされます。コンパイルでエラーが発生すると、エラー・メッセージが表示され、`.rep` ファイルは作成されません。この事態を回避するには、`.rep` ファイルを作成する前に、必ず Reports Builder で「ファイル」⇒「コンパイル」を使用して、ソース・レポートの PL/SQL をコンパイルします。

レポートをテンプレートに変換する場合、レポートのヘッダー・セクション、トレーラ・セクションおよびマージン領域にあるオブジェクトのみがテンプレートに使用されます。メイン・セクションにあるオブジェクトは無視されます。

A.4.29 DUNIT

表 A-31 では、DUNIT キーワードを使用できるコマンドを示します。

表 A-31 DUNIT キーワードを使用できるコマンド

rwclient	rwrn	rwbuilder	rwconverter	rwservlet	rwcgi	rwserver
×	×	×	○	×	×	×

説明 DUNIT を使用して、レポートを変換する宛先のサイズの単位を指定できます。これを指定する場合、DUNIT は、SOURCE レポートのサイズの単位と異なっている必要があります。これを指定しない場合、SOURCE レポートのサイズの単位が使用されます。

構文 DUNIT={CENTIMETER|CHARACTER|INCH|POINT}

値

- CENTIMETER の場合、変換されたレポートは、サイズの単位として最初にセンチメートルを使用します。
- CHARACTER の場合、変換されたレポートは、サイズの単位として最初に文字を使用します。

- INCH の場合、変換されたレポートは、サイズの単位として最初にインチを使用します。
- POINT の場合、変換されたレポートは、サイズの単位として最初にポイントを使用します。

デフォルト NULL

A.4.30 EXPIRATION

表 A-32 では、EXPIRATION キーワードを使用できるコマンドを示します。

表 A-32 EXPIRATION キーワードを使用できるコマンド

rwclient	rwrn	rwbuilder	rwconverter	rwervlet	rwcgi	rwserver
○	×	×	×	○	○	×

説明 EXPIRATION を使用して、レポート出力が削除されるまでキャッシュにとどまることのできる期間を定義します。

重複ジョブの検出の詳細は、[8.8 項「キャッシュからのレポート出力の再使用」](#)（第 8 章）を参照してください。

構文 EXPIRATION=time_string

デフォルト なし

値 時刻文字列には、次のいずれかの形式を使用できます。

- `n{unit}` は、オプションの単位を持つ数値の場合に使用します。`unit` には、分 (s)、時間 (s) または日 (s) を指定できます。単位を指定しない場合は、デフォルトの分 (s) が使用されます。
- `{Mon DD, YYYY} hh:mi:ss am|pm {timezone}` は、日付 / 時刻形式の場合に使用します。日付情報はオプションです。日付を指定しない場合は、本日の日付が使用されます。タイムゾーンもオプションです。タイムゾーンを指定しない場合は、**Reports Server** のタイムゾーンが使用されます。日付 / 時刻には必ず米国のロケールを使用します。この形式は、Java の `DateFormat.MEDIUM` タイプの定義と同じです。

A.4.31 EXPIREDAYS

表 A-33 では、EXPIREDAYS キーワードを使用できるコマンドを示します。

表 A-33 EXPIREDAYS キーワードを使用できるコマンド

rwclient	rwrn	rwbuilder	rwconverter	rwervlet	rwcgi	rwserver
×	×	×	×	○	○	×

説明 EXPIREDAYS を使用して、Oracle Portal に送信されたレポート出力が期限切れになるまでの日数を指定できます。

構文 EXPIREDAYS={PERMANENT|1 day|2 days|3 days|7 days|14 days|31 days|60 days|90 days|120 days}

値 PERMANENT（期限切れなし） |1 day|2 days|3 days|7 days|14 days|31 days|60 days|90 days|120 days

デフォルト なし

使用上の注意 レポート出力を Oracle Portal に送信する際、このキーワードの使用はオプションです。関連するキーワードは、[CONTENTAREA](#)、[EXPIREDAYS](#)、[ITEMTITLE](#)、[OUTPUTFOLDER](#)、[OUTPUTPAGE](#)、[PAGEGROUP](#)、[REPLACEITEM](#)、[SCHEDULE](#)、[SITENAME](#)、[STATUSFOLDER](#) および [STATUSPAGE](#) です。

A.4.32 EXPRESS_SERVER

表 A-34 では、EXPRESS_SERVER キーワードを使用できるコマンドを示します。

表 A-34 EXPRESS_SERVER キーワードを使用できるコマンド

rwclient	rwrn	rwbuilder	rwconverter	rwervlet	rwcgi	rwserver
○	○	○	×	○	○	×

説明 EXPRESS_SERVER を使用して、接続する OLAP Server を指定できます。

構文 EXPRESS_SERVER="server=[server]/domain=[domain]/user=[userid]/password=[passwd] "

RAM の構文 EXPRESS_SERVER="server=[server]/domain=[domain]/user=[userid]/password=[passwd]/ramuser=[ramuserid]/rampassword=[rampasswd]/ramexpressid=[ramexpid]/ramserverscript=[ramsscript]/rammasterdb=[ramdb]/ramconnecttype=[ramconn] "

値 二重引用符 (") で囲まれた有効な接続文字列を指定します。

server	OLAP Server 文字列 (たとえば ncacn_ip_tcp:olap2-pc/sl=x/st=x/ct=x/sv=x/)。サーバー文字列の詳細は次を参照してください。
domain	OLAP Server のドメイン。
user	OLAP Server にログオンするためのユーザー ID。
password	そのユーザー ID に対するパスワード。
ramuser	RDBMS にログインするためのユーザー ID。
rampassword	RDBMS に対するパスワード。
ramexpressid	Oracle Sales Analyzer データベースのユーザー ID。これが必要なのは Oracle Sales Analyzer データベースを使用するときのみです。
ramserverscript	サーバー上のリモート・データベースの構成ファイル (RDC) の完全なファイル名 (フルパスを含む)。このファイルは、コードやデータのデータベースの場所などの情報を指定します。UNC (Universal Naming Convention) 構文を使用すると、同じドライバ文字をその場所にマップすることなく、複数のユーザーが同じ接続を使用してデータにアクセスすることを許可できます。UNC 構文は、 <code>\\ServerName\ShareName\</code> の後に任意のサブフォルダまたはファイルが続きます。
rammasterdb	最初に付加される Relational Access Manager データベースの名前。データベース・ファイル名以外は指定しないでください。このデータベースは OLAP Server の ServerDBPath にあるパス・リストに記載されているディレクトリに常駐している必要があります。ServerDBPath は、Express Configuration Manager の「設定の編集」ダイアログ・ボックスの「ファイル I/O」タブで確認できます。
ramconnecttype	Express 接続の種類。直接接続の場合は必ず 0 を指定してください。

パラメータ サーバー値には、Oracle Express Connection Editor で設定され、接続 (XCF) ファイルに格納されている設定値に対応する 4 つのパラメータが含まれています。これら 4 つのパラメータはすべて必須で、任意の順序で指定できます。表 A-35 に、パラメータとその設定を示します。

表 A-35 EXPRESS_SERVER の server 値とともに使用するパラメータの設定

パラメータ	説明	設定
sl	サーバー・ログイン	-2: ホスト（ドメイン・ログイン） -1: ホスト（サーバー・ログイン） 0: 認証不要 1: ホスト（ドメイン・ログイン）と接続セキュリティ 2: ホスト（ドメイン・ログイン）とコール・セキュリティ 3: ホスト（ドメイン・ログイン）とパケット・セキュリティ 4: ホスト（ドメイン・ログイン）と整合性セキュリティ 5: ホスト（ドメイン・ログイン）とプライバシー・セキュリティ 注意：Windows NT ではすべての設定値を使用します。 UNIX システムでは、設定値 0、-1 および -2 のみを使用します。 これらの設定値の情報は、Express Connection Editor の Help システムを参照してください。
st	サーバーの種類	:1: OLAP Server
ct	接続の種類	0: Express 接続
sv	サーバー・バージョン	1: Express 6.2 以降

デフォルト なし

使用上の注意

- 文字列の全体が引用符で囲まれるので、必要に応じて（たとえば、ユーザー ID が John Smith である場合など）文字列にスペースを含めることができます。
- 文字列にスラッシュ（/）が必要なときは、エスケープ文字としてスラッシュをもう 1 つ使用する必要があります。たとえば、ドメインがツールあるいはレポートである場合は、コマンドラインは次のようになります。

`EXPRESS_SERVER="server=ncacn_ip_tcp:olap2-pc/sl=0/
st=1/ct=0/sv=1/domain=tools//reports"`
- 文字列の中では一重引用符を使用できます。文字列全体が二重引用符で囲まれているため、特別な扱いは受けません。

A.4.33 FORMSIZE

表 A-36 では、FORMSIZE キーワードを使用できるコマンドを示しています。

表 A-36 FORMSIZE キーワードを使用できるコマンド

rwclient	rwrund	rwbuilder	rwconverter	rwervlet	rwcgi	rwserver
×	×	×	○	×	×	×

説明 FORMSIZE を使用して、変換するレポートのランタイム・パラメータ・フォームを、宛先のサイズの単位 (**DUNIT**) で指定できます。

注意： ランタイム・パラメータ・フォームの詳細は、[PARAMFORM](#) キーワードを参照してください。

構文 FORMSIZE=width x height

値 指定されたサイズの単位で利用できるすべての値

デフォルト なし

使用上の注意 キャラクタ以外の **DUNIT** では、端数を指定するのに小数を使用できます (8.5 x 11 など)。

A.4.34 FROM

表 A-37 では、FROM キーワードを使用できるコマンドを示します。

表 A-37 FROM キーワードを使用できるコマンド

rwclient	rwrund	rwbuilder	rwconverter	rwervlet	rwcgi	rwserver
○	○	×	×	○	○	×

説明 FROM を使用して、電子メール送信者の電子メールアドレスを指定できます。

構文 FROM=someone@foo.com

値 有効な電子メール・アドレス

デフォルト loginid@machine_name

使用上の注意 関連するキーワードは、**BCC**、**CC**、**FROM**、**REPLYTO** および **SUBJECT** です。電子メールの主な宛先を指定するには、**DESNAME** を使用します。

A.4.35 GETJOBID

表 A-38 では、GETJOBID キーワードを使用できるコマンドを示します。

表 A-38 GETJOBID キーワードを使用できるコマンド

rwclient	rwrn	rwbuilder	rwconverter	rwervlet	rwcgi	rwserver
×	×	×	×	○	○	×

説明 GETJOBID を使用して、ジョブ ID[n] の Reports Server ジョブの結果出力を取得できます。

構文 `http://yourwebserver/reports/rwervlet/getjobid[n] [?]
[server=server_name] [&authid=username/password] [&statusformat={html|xml|xmldtd}]`

値 構文を参照してください。

デフォルト なし

使用上の注意 ジョブが正常に終了し、Reports Server キャッシュに存在する必要があります。現在のジョブ・リストを参照するには、SHOWJOBS を使用します。ステータスは、html、xml または xmldtd 形式で返されます。デフォルトは html です。

関連するキーワードは、[SERVER](#)、[AUTHID](#) および [STATUSFORMAT](#) です。

STATUSFORMAT パラメータは、rwervlet に対してのみ使用でき、rwcgi には使用できません。

A.4.36 GETSERVERINFO

表 A-39 では、GETSERVERINFO キーワードを使用できるコマンドを示します。

表 A-39 GETSERVERINFO キーワードを使用できるコマンド

rwclient	rwrn	rwbuilder	rwconverter	rwervlet	rwcgi	rwserver
×	×	×	×	○	×	×

説明 GETSERVERINFO を使用して、Reports Server 情報を表示できます。

構文 `http://yourwebserver/reports/rwervlet/getserverinfo[?]
[server=server_name] [&authid=username/password]`

値 構文を参照してください。

デフォルト なし

使用上の注意 関連するキーワードは、[SERVER](#) および [AUTHID](#) です。

A.4.37 HELP

表 A-40 では、HELP キーワードを使用できるコマンドを示します。

表 A-40 HELP キーワードを使用できるコマンド

rwclient	rwrn	rwbuilder	rwconverter	rwervlet	rwsgi	rwserver
×	×	×	×	○	○	×

説明 HELP キーワードを使用して、ヘルプ・トピックを表示し、rwervlet コマンドとともに使用できる追加コマンドを示します。

構文 `http://yourwebserver/reports/rwervlet/help`

値 構文を参照してください。

デフォルト なし

A.4.38 IGNOREMARGIN

表 A-41 では、IGNOREMARGIN キーワードを使用できるコマンドを示します。

表 A-41 IGNOREMARGIN キーワードを使用できるコマンド

rwclient	rwrn	rwbuilder	rwconverter	rwervlet	rwsgi	rwserver
○	○	○	×	○	○	×

説明 IGNOREMARGIN を使用して、Reports がプリンタのハードウェア・マージンを無視してレポート定義の物理ページ・マージンで指定された値を使用するかどうかを指定できます。

構文 `IGNOREMARGIN={YES|NO}`

値 YES を指定すると、Reports がプリンタのハードウェア・マージンを無視してレポート定義の物理ページ・マージンで指定された値を使用します。NO を指定すると、レポート印刷時、Reports がプリンタのハードウェア・マージンをレポート定義の物理ページ・マージンで指定された値に追加します。

デフォルト NO

A.4.39 INSTALL

表 A-42 では、INSTALL キーワードを使用できるコマンドを示します。

表 A-42 INSTALL キーワードを使用できるコマンド

rwclient	rwrund	rwbuilder	rwconverter	rwservlet	rwcgi	rwserver
×	×	×	×	×	×	○

説明 INSTALL を使用して、Microsoft Windows 上の Reports Server インスタンスをサービスとして構成できます。この引数は UNIX プラットフォームでは無効です。

構文 -INSTALL REPORTS_SERVER_NAME

値 Reports Server インスタンスの有効な名前

デフォルト なし

使用上の注意 INSTALL とともに AUTOSTART キーワードを使用すると、インストール後およびシステム再起動後に Reports Server サービスが自動的に開始されます。

INSTALL とともに BATCH=YES を使用すると、通常インストール時に表示されるプロンプトおよびダイアログが表示されなくなります。

A.4.40 ITEMTITLE

表 A-43 では、ITEMTITLE キーワードを使用できるコマンドを示します。

表 A-43 ITEMTITLE キーワードを使用できるコマンド

rwclient	rwrund	rwbuilder	rwconverter	rwservlet	rwcgi	rwserver
○	○	×	×	○	○	×

説明 ITEMTITLE を使用して、Oracle Portal がレポート出力に使用する表示名を指定できます。Oracle Portal にはこの名前が表示され、レポート出力にリンクされます。

構文 ITEMTITLE="Your output title"

値 任意のテキスト

デフォルト レポートのファイル名

使用上の注意 レポート出力を Oracle Portal に送信する際、このキーワードの使用はオプションです。値に文字スペース単位が含まれる場合や、cgicmd.dat ファイルで引数を指定する場合は、値の前後に引用符を付けます。

関連するキーワードは、[CONTENTAREA](#)、[EXPIREDAYS](#)、[ITEMTITLE](#)、[OUTPUTFOLDER](#)、[OUTPUTPAGE](#)、[PAGEGROUP](#)、[REPLACEITEM](#)、[SCHEDULE](#)、[SITENAME](#)、[STATUSFOLDER](#) および [STATUSPAGE](#) です。

A.4.41 JOBNAME

表 A-44 では、JOBNAME キーワードを使用できるコマンドを示します。

表 A-44 JOBNAME キーワードを使用できるコマンド

rwclient	rwrn	rwbuilder	rwconverter	rservlet	rwsgi	rwserver
○	×	×	×	○	○	×

説明 JOBNAME を使用して、Oracle9iAS Reports Queue Manager に表示されるジョブの名前を指定できます。これはコメントとして扱われ、ジョブの実行にはまったく関係しません。JOBNAME が指定されていないときは、Oracle9iAS Reports Queue Manager ではレポート名がジョブ名として表示されます。

構文 JOBNAME=*string*

値 任意のジョブ名

デフォルト なし

使用上の注意 JOBNAME は、JSP として実行されるジョブでも使用できます。

A.4.42 JOBTYP

表 A-45 では、JOBTYP キーワードを使用できるコマンドを示します。

表 A-45 JOBTYP キーワードを使用できるコマンド

rwclient	rwrn	rwbuilder	rwconverter	rservlet	rwsgi	rwserver
○	×	×	×	○	○	×

説明 JOBTYP を使用して、サーバーによって実行されるジョブのタイプを記述できます。Reports Server に備わっているエンジンがそのジョブ・タイプを処理できれば、任意のジョブ・タイプを入力できます。

構文 JOBTYP= {Reports Server のエンジンで処理可能なジョブ・タイプ }

デフォルト REPORT

A.4.43 KILLJOBID

表 A-46 では、KILLJOBID キーワードを使用できるコマンドを示します。

表 A-46 KILLJOBID キーワードを使用できるコマンド

rwclient	rwrn	rwbuilder	rwconverter	rwervlet	rwcgi	rwserver
×	×	×	×	○	○	×

説明 KILLJOBID を使用して、ジョブ ID[n] の Reports Server ジョブを取り消すことができます。

構文 http://yourwebserver/reports/rwervlet/killjobid[n] [?] [server=server_name] [&authid=username/password] [&statusformat={html|xml|xmldtd}]

値 構文を参照してください。

デフォルト なし

使用上の注意 ジョブは現行である (キューに入っているかスケジュールされている) 必要があります。現在のジョブ・リストを参照するには、SHOWJOBS を使用します。STATUSFORMAT は html、xml または xmldtd に設定できます。指定した形式でステータスが返されます。デフォルトは html です。

関連するキーワードは、SHOWJOBS、SERVER、AUTHID および STATUSFORMAT です。STATUSFORMAT パラメータは、rwervlet に対してのみ使用でき、rwcgi には使用できません。

A.4.44 LONGCHUNK

表 A-47 では、LONGCHUNK キーワードを使用できるコマンドを示します。

表 A-47 LONGCHUNK キーワードを使用できるコマンド

rwclient	rwrn	rwbuilder	rwconverter	rwervlet	rwcgi	rwserver
○	○	○	×	○	○	×

説明 LONGCHUNK は、Oracle9iDS Reports Builder で LONG 列値を取り出す変更分のサイズ (KB) です。LONG の値を検索する際に、メモリー容量の制限から一度にすべてを検索するのではなく、少しずつ検索する方がよい場合があります。LONGCHUNK は、Oracle データベースのみに適用されます。

構文 LONGCHUNK=*n*

値 1 ～ 9999 までの数字を使用します (位取りのカンマや小数点などは使用しないでください)。オペレーティング・システムによっては、上限がこれより低い場合もあります。

デフォルト 10

使用上の注意 LONGCHUNK は、JSP として実行されるジョブでも使用できます。

A.4.45 MODE

表 A-48 では、MODE キーワードを使用できるコマンドを示します。

表 A-48 MODE キーワードを使用できるコマンド

rwclient	rwrn	rwbuilder	rwconverter	rwervlet	rwsgi	rwserver
○	○	×	×	○	○	×

説明 MODE を使用して、レポートをキャラクタ・モードで実行するか、ビットマップで実行するかを指定できます。

構文 MODE={BITMAP|CHARACTER|DEFAULT}

値 次の値が適用されます。

- BITMAP
- CHARACTER
- DEFAULT (BITMAP です)

デフォルト DEFAULT

A.4.46 MODULE|REPORT

表 A-49 では、MODULE|REPORT キーワードを使用できるコマンドを示します。

表 A-49 MODULE|REPORT キーワードを使用できるコマンド

rwclient	rwrn	rwbuilder	rwconverter	rwservlet	rwcgi	rwserver
○	○	○	×	○	○	×

説明 MODULE または REPORT を使用して、実行するレポートの名前を指定できます。

構文 REPORT|MODULE=*runfile*

値 任意の有効な実行ファイル（つまり、拡張子が REP、RDF、JSP または XML のファイル）を指定します。ファイル拡張子を入力しないと、Oracle9iAS Reports Runtime は REP 拡張子、RDF 拡張子、JSP 拡張子、拡張子なしの順にファイルを検索します。Oracle9iAS Reports Runtime では、ファイル名の前にディレクトリ・パスがない場合、ファイルの検索に REPORTS_PATH 検索順序が使用されます。

デフォルト なし

A.4.47 NONBLOCKSQL

表 A-50 では、NONBLOCKSQL キーワードを使用できるコマンドを示します。

表 A-50 NONBLOCKSQL キーワードを使用できるコマンド

rwclient	rwrn	rwbuilder	rwconverter	rwservlet	rwcgi	rwserver
○	○	○	×	○	○	×

説明 NONBLOCKSQL を使用して、Reports Runtime でデータベースからデータをフェッチしている間に、他のプログラムを実行できるかどうかを指定できます。

構文 NONBLOCKSQL={YES|NO}

値 YES は、データをフェッチしているときに他のプログラムを実行できます。NO は、データをフェッチしているときは他のプログラムを実行できません。

デフォルト YES

使用上の注意 NONBLOCKSQL は、JSP として実行されるジョブでも使用できます。

A.4.48 NOTIFYFAILURE

表 A-51 では、NOTIFYFAILURE キーワードを使用できるコマンドを示します。

表 A-51 NOTIFYFAILURE キーワードを使用できるコマンド

rwclient	rwrund	rwbuilder	rwconverter	rwervlet	rwsgi	rwserver
○	×	×	×	○	○	×

説明 NOTIFYFAILURE を使用して、レポート要求が失敗した場合に電子メールで通知する宛先を指定できます。このキーワードは、通知クラスを使用するよう Reports Server を構成するときに使用します。詳細は、第 3 章「[Oracle9iAS Reports Services の構成](#)」の通知に関する説明を参照してください。

構文 NOTIFYFAILURE={name1@mycompany.com,name2@mycompany.com}

値 1 つ以上の有効な電子メール・アドレス

デフォルト なし

使用上の注意 通知電子メールの本文に使用されるデフォルトの通知電子メールのテンプレートが、Oracle9iAS のインストールに含まれています。NOTIFYFAILURE テンプレートの名前は failnote.txt、位置は %ORACLE_HOME%\reports\template です。

NOTIFYFAILURE は、JSP として実行されるジョブでも使用できます。

A.4.49 NOTIFYSUCCESS

表 A-52 では、NOTIFYSUCCESS キーワードを使用できるコマンドを示します。

表 A-52 NOTIFYSUCCESS キーワードを使用できるコマンド

rwclient	rwrund	rwbuilder	rwconverter	rwervlet	rwsgi	rwserver
○	×	×	×	○	○	×

説明 NOTIFYSUCCESS を使用して、レポート要求が成功した場合に電子メールで通知する宛先を指定できます。このキーワードは、通知クラスを使用するよう Reports Server を構成するときに使用します。詳細は、第 3 章「[Oracle9iAS Reports Services の構成](#)」の通知に関する説明を参照してください。

構文 NOTIFYSUCCESS={name1@mycompany.com,name2@mycompany.com}

値 1 つ以上の有効な電子メール・アドレス

デフォルト なし

使用上の注意 通知電子メールの本文に使用されるデフォルトの通知電子メールのテンプレートが、Oracle9iAS のインストールに含まれています。NOTIFYSUCCESS テンプレートの名前は succnote.txt、位置は %ORACLE_HOME%\reports\template です。

NOTIFYSUCCESS は、JSP として実行されるジョブでも使用できます。

A.4.50 NUMBERFORMATMASK

表 A-53 では、NUMBERFORMATMASK キーワードを使用できるコマンドを示します。

表 A-53 NUMBERFORMATMASK キーワードを使用できるコマンド

rwclient	rwrun	rwbuilder	rwconverter	rwervlet	rwcgi	rwserver
○	○	×	×	○	○	×

説明 NUMBERFORMATMASK を使用して、数値がデリミタ付きのレポート出力でどのように表示されるかを指定します。

構文 NUMBERFORMATMASK=*mask*

値 任意の数値書式マスク

デフォルト なし

使用上の注意 このキーワードを使用できるのは、DESFORMAT=DELIMITED と指定しているときのみです。

注意： 有効な NUMBERFORMATMASK 値は、Report Builder のオンライン・ヘルプのトピック「数値書式マスク構文」を参照してください。

NUMBERFORMATMASK は、JSP として実行されるジョブでも使用できます。

A.4.51 ONFAILURE

表 A-54 では、ONFAILURE キーワードを使用できるコマンドを示します。

表 A-54 ONFAILURE キーワードを使用できるコマンド

rwclient	rwrun	rwbuilder	rwconverter	rwervlet	rwcgi	rwserver
○	○	○	×	○	○	×

説明 ONFAILURE を使用して、エラーが発生してレポートが完了されなかった場合に COMMIT または ROLLBACK を実行させるかどうかを指定できます。

構文 ONFAILURE={COMMIT|ROLLBACK|NOACTION}

値 COMMIT は、レポートが完了しなかった場合に COMMIT を実行します。ROLLBACK は、レポートが完了しなかった場合に ROLLBACK を実行します。NOACTION では、レポートが失敗しても何も行いません。

デフォルト ユーザー ID が指定されている場合は、ROLLBACK です。ユーザー ID の指定がなく、外部ソース（Oracle9iDS Forms Services など）から呼び出される場合は NOACTION です。

使用上の注意 ONFAILURE の COMMIT または ROLLBACK は、レポートが失敗した後に実行されます。これに先だって、他の COMMIT および ROLLBACK が発生する場合もあります。詳細は、[READONLY](#) コマンドの項を参照してください。

ONFAILURE は、JSP として実行されるジョブでも使用できます。

A.4.52 ONSUCCESS

[表 A-55](#) では、ONSUCCESS キーワードを使用できるコマンドを示します。

表 A-55 ONSUCCESS キーワードを使用できるコマンド

rwclient	rwrwn	rwbuilder	rwconverter	rwervlet	rwcgi	rwserver
○	○	○	×	○	○	×

説明 ONSUCCESS を使用して、レポートの実行が終了した時点で COMMIT または ROLLBACK のどちらを実行するかを指定できます。

構文 ONSUCCESS={COMMIT|ROLLBACK|NOACTION}

値 COMMIT は、レポート完了時に COMMIT を実行します。ROLLBACK は、レポート完了時に ROLLBACK を実行します。NOACTION の場合、レポートが終了しても何も行われません。

デフォルト ユーザー ID が指定されている場合は COMMIT です。ユーザー ID の指定がなく、外部ソース（Oracle9iDS Forms Services など）から呼び出される場合は NOACTION です。

使用上の注意 ONSUCCESS に対する COMMIT または ROLLBACK は、レポートのトリガーが発行されてから実行されます。その他の COMMIT および ROLLBACK は、これより前に発生する場合があります。詳細は、[READONLY](#) コマンドの項を参照してください。

ONSUCCESS は、JSP として実行されるジョブでも使用できます。

A.4.53 ORIENTATION

表 A-56 では、ORIENTATION キーワードを使用できるコマンドを示します。

表 A-56 ORIENTATION キーワードを使用できるコマンド

rwclient	rwrn	rwbuilder	rwconverter	rwservlet	rwcgi	rwserver
○	○	×	×	○	○	×

説明 ORIENTATION は、レポートの各ページのプリント方向をコントロールします。

構文 ORIENTATION={DEFAULT|LANDSCAPE|PORTRAIT}

値 DEFAULT の場合、現行のプリンタの方向設定を使用します。LANDSCAPE は横方向です (長辺が上下になる配置)。PORTRAIT は縦方向です (短辺が上下になる配置)。

デフォルト DEFAULT

使用上の注意

- キャラクタ・モードのレポートに対して ORIENTATION=LANDSCAPE を指定する場合、プリンタ定義ファイルに横方向を指定する句が含まれていることが必要です。
- Motif 上の PCL プリンタに出力する場合はサポートされていません。

A.4.54 OUTPUTFOLDER

表 A-57 では、OUTPUTFOLDER キーワードを使用できるコマンドを示します。

表 A-57 OUTPUTFOLDER キーワードを使用できるコマンド

rwclient	rwrn	rwbuilder	rwconverter	rwservlet	rwcgi	rwserver
○	○	×	×	○	○	×

説明 OUTPUTFOLDER を使用して、レポート出力の配信先となる Oracle Portal フォルダの名前を指定できます。このキーワードは、Oracle Portal の旧バージョン (WebDB 2.2 および Oracle Portal 3.0.9) に対する下位互換性を確保するために維持されています。Oracle9iAS Portal バージョン 2.0 では、[OUTPUTPAGE](#) を使用してください。

構文 OUTPUTFOLDER=Oracle_Reports_Output

値 Oracle9iAS Portal で使用されているフォルダ名

デフォルト Oracle_Reports_Output

使用上の注意 このキーワードの値は大文字と小文字を区別します。レポート出力を Oracle Portal に配信する際、このキーワードの使用は必須です。値に文字スペース単位が含まれる場合や、cgicmd.dat ファイルで引数を指定する場合は、値の前後に引用符を付けます。

関連するキーワードは、[CONTENTAREA](#)、[EXPIREDAYS](#)、[ITEMTITLE](#)、[OUTPUTFOLDER](#)、[OUTPUTPAGE](#)、[PAGEGROUP](#)、[REPLACEITEM](#)、[SCHEDULE](#)、[SITENAME](#)、[STATUSFOLDER](#) および [STATUSPAGE](#) です。

ページやページ・グループなどの Oracle Portal オブジェクトには、表示名と内部名という 2 つの名前があります。Oracle Portal 出力で使用するオブジェクトを Oracle Portal 内で作成する際、Oracle Portal のオンライン・ヘルプで指定された内部名の命名規則に従って、内部名と表示名を同じにします。これにより、Reports コマンドラインで Portal 関連のキーワードに値を指定する際、どちらの名前を指定すればよいか考慮する必要がなくなります。

A.4.55 OUTPUTPAGE

表 A-58 では、OUTPUTPAGE キーワードを使用できるコマンドを示します。

表 A-58 OUTPUTPAGE キーワードを使用できるコマンド

rwclient	rwrn	rwbuilder	rwconverter	rwservlet	rwcgi	rwserver
○	○	×	×	○	○	×

説明 OUTPUTPAGE を使用して、レポート出力の配信先となる Oracle Portal ページの名前を指定できます (Oracle9iAS Portal バージョン 2.0 よりも古いバージョンとの互換性については、[「OUTPUTFOLDER」](#)を参照してください)。

構文 OUTPUTPAGE=Oracle_Reports_Output

値 Oracle Portal で使用されているページ名

デフォルト Oracle_Reports_Output

使用上の注意 このキーワードの値は大文字と小文字を区別します。レポート出力を Oracle Portal に配信する際、このキーワードの使用はオプションです。出力ページを指定しない場合、Oracle Reports Output という名前のデフォルト・ページが Oracle9iAS Portal により作成されます。

値に文字スペース単位が含まれる場合や、cgicmd.dat ファイルで引数を指定する場合は、値の前後に引用符を付けます。

レポート出力の Oracle Portal への配信に関連するキーワードは、CONTENTAREA、EXPIREDAYS、ITEMTITLE、OUTPUTFOLDER、OUTPUTPAGE、PAGEGROUP、REPLACEITEM、SCHEDULE、SITENAME、STATUSFOLDER および STATUSPAGE です。

ページやページ・グループなどの Oracle Portal オブジェクトには、表示名と内部名という 2 つの名前があります。Oracle Portal 出力で使用するオブジェクトを Oracle Portal 内で作成する際、Oracle Portal のオンライン・ヘルプで指定された内部名の命名規則に従って、内部名と表示名を同じにします。これにより、Reports コマンドラインで Portal 関連のキーワードに値を指定する際、どちらの名前を指定すればよいか考慮する必要がなくなります。

A.4.56 OVERWRITE

表 A-59 では、OVERWRITE キーワードを使用できるコマンドを示します。

表 A-59 OVERWRITE キーワードを使用できるコマンド

rwclient	rwrn	rwbuilder	rwconverter	rwervlet	rwcgi	rwserver
×	×	×	○	×	×	×

説明 OVERWRITE を使用して、既存のファイルを、変換済みのファイルで上書きするかどうかを指定できます。

構文 OVERWRITE={YES|NO|PROMPT}

値

- YES の場合、rwconverter は同名の既存ファイルを自動的にすべて上書きします。
- NO の場合、同名の既存ファイルがあっても、レポートを変換せず、警告メッセージを表示します。
- PROMPT の場合、既存ファイルを上書きする前に、確認を要求します。

デフォルト NO

A.4.57 P_AVAILABILITY

表 A-60 では、P_AVAILABILITY キーワードを使用できるコマンドを示します。

表 A-60 P_AVAILABILITY キーワードを使用できるコマンド

rwclient	rwrn	rwbuilder	rwconverter	rwervlet	rwsgi	rwserver
×	×	×	○	×	×	×

説明 P_AVAILABILITY は使用可能日カレンダーの名前で、指定されたレポートがいつ処理できるようになるかを決定します。このキーワードが使用されるのは、DTYPE=REGISTER の場合のみです。

構文 P_AVAILABILITY=calendar_name

値 有効な使用可能日カレンダー名

デフォルト なし

使用上の注意 使用可能日カレンダーは、この SQL*PLUS スクリプトを実行する前に、Oracle9iAS Portal に登録されている必要があります。登録されていない場合、無効なパッケージが作成されることがあります。

A.4.58 P_DESCRIPTION

表 A-61 では、P_DESCRIPTION キーワードを使用できるコマンドを示します。

表 A-61 P_DESCRIPTION キーワードを使用できるコマンド

rwclient	rwrn	rwbuilder	rwconverter	rwervlet	rwsgi	rwserver
×	×	×	○	×	×	×

説明 P_DESCRIPTION は、レポートに関する追加情報を提供するテキストです。このキーワードが使用されるのは、DTYPE=REGISTER の場合のみです。

構文 P_DESCRIPTION=DESCRIPTION_TEXT

値 任意のテキスト文字列

デフォルト なし

A.4.59 P_FORMATS

表 A-62 では、P_FORMATS キーワードを使用できるコマンドを示します。このキーワードが使用されるのは、DTYPE=REGISTER の場合のみです。

表 A-62 P_FORMATS キーワードを使用できるコマンド

rwclient	rwrn	rwbuilder	rwconverter	rwervlet	rwcgi	rwserver
×	×	×	○	×	×	×

説明 P_FORMATS は、指定されたレポートに許可された宛先フォーマットです。このキーワードが使用されるのは、DTYPE=REGISTER の場合のみです。

構文 P_FORMATS=(HTMLCSS,PDF,...)

値 有効な宛先タイプ (HTML など) または、名前をカンマで区切り、括弧で囲んだ有効な宛先タイプのリスト ((HTMLCSS,PDF,RTF) など) を指定します。

デフォルト なし

A.4.60 P_NAME

表 A-63 では、P_NAME キーワードを使用できるコマンドを示します。

表 A-63 P_NAME キーワードを使用できるコマンド

rwclient	rwrn	rwbuilder	rwconverter	rwervlet	rwcgi	rwserver
×	×	×	○	×	×	×

説明 P_NAME は、Oracle9iAS Portal で表示されるレポート名です。このキーワードが使用されるのは、DTYPE=REGISTER の場合のみです。

構文 P_NAME=REPORT_NAME

値 任意のレポート名

デフォルト P_NAME を指定しない場合、PL/SQL ファンクションにはレポート定義ファイル名が入力されます。

使用上の注意 P_NAME は、Oracle9iAS Portal に登録する各レポート定義ファイルに対して同じレポート名を使用する場合のみ指定します。この引数は通常、空白にします。

レポート名の先頭には数値を使用できません（たとえば、401K_report は無効なファイル名ですが、my_401K_report は有効です）。

A.4.61 P_OWNER

表 A-64 では、P_OWNER キーワードを使用できるコマンドを示します。

表 A-64 P_OWNER キーワードを使用できるコマンド

rwclient	rwrund	rwbuilder	rwconverter	rwservlet	rwcgi	rwserver
×	×	×	○	×	×	×

説明 P_OWNER は、レポートのパッケージを所有する Oracle9iAS Portal スキーマで、レポート定義ファイルの登録時に作成されます。このキーワードが使用されるのは、DTYPE=REGISTER の場合のみです。

構文 P_OWNER=PORTAL_SCHEMA_NAME

値 有効な Oracle9iAS Portal スキーマ名

デフォルト SQL*PLUS スクリプト・ファイル実行時にログインする Oracle9iAS Portal スキーマ名

A.4.62 P_PFORMTEMPLATE

表 A-65 では、P_PFORMTEMPLATE キーワードを使用できるコマンドを示します。

表 A-65 P_PFORMTEMPLATE キーワードを使用できるコマンド

rwclient	rwrund	rwbuilder	rwconverter	rwservlet	rwcgi	rwserver
×	×	×	○	×	×	×

説明 P_PFORMTEMPLATE は、ランタイム・パラメータ・フォームのスタイルを決定する Oracle9iAS Portal テンプレートの名前です。このキーワードが使用されるのは、DTYPE=REGISTER の場合のみです。

構文 P_PFORMTEMPLATE=TEMPLATE_NAME

値 有効な Oracle9iAS Portal テンプレート名

デフォルト なし

A.4.63 P_PRINTERS

表 A-66 では、P_PRINTERS キーワードを使用できるコマンドを示します。

表 A-66 P_PRINTERS キーワードを使用できるコマンド

rwclient	rwrund	rwbuilder	rwconverter	rwservlet	rwcgi	rwserver
×	×	×	○	×	×	×

説明 P_PRINTERS は、指定されたレポートに許可されたプリンタです。このキーワードが使用されるのは、DTYPE=REGISTER の場合のみです。

構文 P_PRINTERS=(PRT1,PRT2,...)

値 有効なプリンタ (PRT1 など) または、名前をカンマで区切り、括弧で囲んだ有効なプリンタのリスト ((PRT1,PRT2,PRT3) など) を指定します。

デフォルト なし

使用上の注意 プリンタへのアクセスは、この SQL*Plus スクリプトを実行する前に、Oracle9iAS Portal に登録されている必要があります。

A.4.64 P_PRIVILEGE

表 A-67 では、P_PRIVILEGE キーワードを使用できるコマンドを示します。

表 A-67 P_PRIVILEGE キーワードを使用できるコマンド

rwclient	rwrund	rwbuilder	rwconverter	rwservlet	rwcgi	rwserver
×	×	×	○	×	×	×

説明 P_PRIVILEGE は、選択したレポートを実行するアクセス権限を持つユーザーまたはロールです。このキーワードが使用されるのは、DTYPE=REGISTER の場合のみです。

構文 P_PRIVILEGE=(SCOTT,JABERS,PMARTIN,...)

値 Oracle9iAS Portal が認識できるユーザー名またはロール (SCOTT など)、または、名前をカンマで区切り、括弧で囲んだユーザー名またはロールのリスト ((SCOTT,JABERS,PMARTIN) など) を指定します。

デフォルト なし

A.4.65 P_SERVERS

表 A-68 では、P_SERVERS キーワードを使用できるコマンドを示します。

表 A-68 P_SERVERS キーワードを使用できるコマンド

rwclient	rwrund	rwbuilder	rwconverter	rwervlet	rwsgi	rwserver
×	×	×	○	×	×	×

説明 P_SERVERS は、レポートを実行できる制限付き Reports Server の名前です。このキーワードが使用されるのは、DTYPE=REGISTER の場合のみです。

構文 P_SERVERS=(repserver1,repserver2,...)

値 Reports Server の有効な TNS 名 (repserver など) または、名前をカンマで区切り、括弧で囲んだ Reports Server の有効な TNS 名のリスト ((repserver,acct_server,sales_server) など) を指定します。

デフォルト なし

使用上の注意 Reports Server へのアクセスが、すでに Oracle9iAS Portal に登録されている必要があります。

A.4.66 P_TRIGGER

表 A-69 では、P_TRIGGERS キーワードを使用できるコマンドを示します。

表 A-69 P_TRIGGER キーワードを使用できるコマンド

rwclient	rwrund	rwbuilder	rwconverter	rwervlet	rwsgi	rwserver
×	×	×	○	×	×	×

説明 P_TRIGGER は、パラメータ値をコマンドラインで指定する際およびランタイム・パラメータ・フォームを確定する際に実行される PL/SQL ファンクションです。ファンクションは、ブール値（TRUE または FALSE）を戻す必要があります。次に例を示します。

```
P_TRIGGER=Is begin IF UPPER(DESTYPE) = 'PRINTER' AND EMPNAME = 'SMITH' THEN
RETURN(TRUE); ELSE RETURN(FALSE); END IF; end;
```

このキーワードが使用されるのは、DTYPE=REGISTER の場合のみです。

構文 P_TRIGGER=PLSQL_FUNCTION

値 ブール値を戻す有効な PL/SQL ファンクション

デフォルト なし

A.4.67 P_TYPES

表 A-70 では、P_TYPES キーワードを使用できるコマンドを示します。

表 A-70 P_TYPES キーワードを使用できるコマンド

rwclient	rwrun	rwbuilder	rwconverter	rwservlet	rwcgi	rwserver
×	×	×	○	×	×	×

説明 P_TYPES は、指定されたレポートに許可された宛先タイプです。このキーワードが使用されるのは、DTYPE=REGISTER の場合のみです。

構文 P_TYPES=(CACHE,MAIL,...)

値 有効な宛先タイプ（CACHE など）または、名前をカンマで区切り、括弧で囲んだ有効な宛先タイプのリスト（(CACHE,MAIL,PRINTER) など）を指定します。

デフォルト なし

A.4.68 PAGEGROUP

表 A-71 では、PAGEGROUP キーワードを使用できるコマンドを示します。

表 A-71 PAGEGROUP キーワードを使用できるコマンド

rwclient	rwrun	rwbuilder	rwconverter	rwservlet	rwcgi	rwserver
○	○	×	×	○	○	×

説明 PAGEGROUP を使用して、レポート出力の配信先となる Oracle Portal ページ・グループの名前を指定できます。WebDB 2.2 では、かわりに [SITENAME](#) を使用してください。Oracle Portal 3.0 では、かわりに [CONTENTAREA](#) を使用してください。Oracle9iAS Portal バージョン 2.0 以降では、PAGEGROUP を使用してください。

このパラメータを使用するには、ページ・グループを事前に Oracle Portal で作成しておく必要があります。

構文 PAGEGROUP="Name of Oracle Portal page group"

値 Oracle Portal で使用されているページ・グループ

デフォルト なし

使用上の注意 レポート出力を Oracle Portal に配信する際、このキーワードの使用は必須です。値に文字スペース単位が含まれる場合や、cgicmd.dat ファイルで引数を指定する場合は、値の前後に引用符を付けます。

関連するキーワードは、[CONTENTAREA](#)、[EXPIREDAYS](#)、[ITEMTITLE](#)、[OUTPUTFOLDER](#)、[OUTPUTPAGE](#)、[PAGEGROUP](#)、[REPLACEITEM](#)、[SCHEDULE](#)、[SITENAME](#)、[STATUSFOLDER](#) および [STATUSPAGE](#) です。

ページやページ・グループなどの Oracle Portal オブジェクトには、表示名と内部名という 2 つの名前があります。Oracle Portal 出力で使用するオブジェクトを Oracle Portal 内で作成する際、Oracle Portal のオンライン・ヘルプで指定された内部名の命名規則に従って、内部名と表示名を同じにします。これにより、Reports コマンドラインで Portal 関連のキーワードに値を指定する際、どちらの名前を指定すればよいか考慮する必要がなくなります。

A.4.69 PAGESIZE

[表 A-72](#) では、PAGESIZE キーワードを使用できるコマンドを示します。

表 A-72 PAGESIZE キーワードを使用できるコマンド

rwclient	rwrn	rwbuilder	rwconverter	rwservlet	rwcgi	rwserver
○	○	○	○	○	○	×

説明 PAGESIZE を使用して、物理ページの大きさ（プリンタが出力するページのサイズ）を設定できます。ページはレポートが入る大きさである必要がありますので注意してください。たとえば、レポートのフレームがページ・ディメンションを超えるサイズになっている場合、レポートは実行されません。

構文 PAGESIZE=width x height

値 フォームに対する任意の有効なページ・ディメンションを指定します（ページ幅×ページ高さ:それぞれ0より大きい数）。最大幅・高さは、**Reports Builder** に設定されている測定単位によって異なります（「編集」⇒「作業環境」⇒「一般」タブ）。インチでは、最大幅・高さは512インチです。cmでは1312cmです。パイカでは36,864パイカです。

デフォルト ビットマップの場合は8.5 × 11 インチです。キャラクタ・モードの場合は80 × 66 文字です。レポートがキャラクタ・モードに対して設計されていて、ビットマップで実行あるいは変換されている場合、ページ・サイズが指定されていないときは次の式を使用して決定されます。（デフォルトのページ・サイズ* 文字ページのサイズ） / デフォルトの文字のページ・サイズ。たとえば、文字ページのサイズが80 × 20 の場合、ビットマップのページは次のサイズになります。
$$((8.5 * 80)/80) \times ((11 * 20)/66) = (680/80) \times (220/66) = 8.5 \times 3.33$$

使用上の注意

- プリンタによっては、物理ページの印刷可能な面積に制限があります。たとえば、プリンタが受け入れる用紙は8.5 × 11 インチであっても、実際にプリントできる面積は8 × 10.5 インチである可能性もあります。**Reports Builder** で、使用するプリンタが許容する印刷可能面積を超えるページ幅とページ高さを定義すると、レポートを出力した際にクリッピングが生じる可能性があります。クリッピングを防ぐには、プリンタの印刷可能領域を拡大するか（オペレーティング・システムで可能な場合）、ページ幅×ページの高さをページの印刷可能な領域のサイズに設定します。
- レター・サイズは、8.5 × 11 インチです。A4 サイズは、210mm × 297mm です。
- PAGESIZE キーワードを使用すると、レポート定義のページ・ディメンションはその値によって上書きされます。
- コマンドラインに入力された PAGESIZE 値は、ランタイム・パラメータ・フォームに入力された PAGESIZE 値によって上書きされます。

A.4.70 PAGESTREAM

表 A-73 では、PAGESTREAM キーワードを使用できるコマンドを示します。

表 A-73 PAGESTREAM キーワードを使用できるコマンド

rwclient	rwrn	rwbuilder	rwconverter	rwservlet	rwcgi	rwserver
○	○	×	×	○	○	×

説明 PAGESTREAM は、レポートが HTML あるいは HTMLCSS としてフォーマットされている場合に、次のいずれかで設定されているナビゲーション・コントロールを使用して、そのレポートのページ・ストリームを可能または不可能にします。

- レポート・プロパティ・パレットのページ・ナビゲーション・コントロール値プロパティおよびページ・ナビゲーション・コントロール・タイプ・プロパティ

- Before Report トリガーの PL/SQL (SRW.SET_PAGE_NAVIGATION_HTML)

構文 PAGESREAM={YES|NO}

値 YES の場合、ページはストリーム処理されます。NO の場合、ページ・ストリームなしでレポートが出力されます。

デフォルト NO

A.4.71 PARAMFORM

表 A-74 では、PARAMFORM キーワードを使用できるコマンドを示します。

表 A-74 PARAMFORM キーワードを使用できるコマンド

rwclient	rwrwn	rwbuilder	rwconverter	rwservlet	rwcgi	rwserver
○	×	×	×	○	○	×

説明 PARAMFORM を使用して、レポートを CGI またはサーブレット経由で実行するときにランタイム・パラメータ・フォームを表示するかどうかを指定します。PARAMFORM は、Web ソース・レポートではなく、ペーパー・レイアウト・レポートにパラメータを指定するために使用されます。

構文 PARAMFORM=YES|NO|HTML

値 YES の場合、パラメータ・フォームが表示されます。NO の場合、パラメータ・フォームが表示されません。HTML の場合、パラメータ・フォームが HTML 形式で表示されます。

デフォルト NO

使用上の注意 レポートを Oracle Portal 環境で実行するときは、このキーワードを使用しないでください。これは、Oracle Portal ではレポートのランタイム・パラメータ・フォームの設定が許可されており、この設定が PARAMFORM キーワードで指定したフォームと競合する可能性があるためです。

A.4.72 PARSEQUERY

表 A-75 では、PARSEQUERY キーワードを使用できるコマンドを示します。

表 A-75 PARSEQUERY キーワードを使用できるコマンド

rwclient	rwrwn	rwbuilder	rwconverter	rwservlet	rwcgi	rwserver
×	×	×	×	○	○	×

説明 PARSEQUERY を使用して、rwservlet 問合せを解析し、作成された Reports Server コマンドラインを表示できます。

構文 `http://yourwebserver/reports/rwservlet/parsequery[?]
[server=servername] [&authid=username/password] query_string`

値 構文を参照してください。

デフォルト なし

A.4.73 PDFCOMP

表 A-76 では、PDFCOMP キーワードを使用できるコマンドを示します。

表 A-76 PDFCOMP キーワードを使用できるコマンド

rwclient	rwrund	rwbuilder	rwconverter	rwservlet	rwcgi	rwserver
○	○	×	×	○	○	×

説明 PDFCOMP を使用して、PDF 出力を圧縮するかどうかを指定できます。

構文 `PDFCOMP={any value 0 through 9} OR {YES|NO}`

値 0 ～ 9、YES (6) または NO (0) のうち任意の値を指定します。0 の場合、PDF 出力は圧縮されません。1 ～ 9 の場合、PDF 出力が圧縮され、ユーザーが圧縮レベルを制御できます。YES は圧縮レベル 6 と同じです。NO は圧縮レベル 0 です。

デフォルト 6

A.4.74 PDFEMBED

表 A-77 では、PDFEMBED キーワードを使用できるコマンドを示します。

表 A-77 PDFEMBED キーワードを使用できるコマンド

rwclient	rwrund	rwbuilder	rwconverter	rwservlet	rwcgi	rwserver
○	○	×	×	○	○	×

説明 PDFEMBED を使用して、uifont.ali で指定した Type1 PostScript フォント・ファイルを Reports が PDF 出力に組み込むかどうかを指定できます。

構文 `PDFEMBED={YES|NO}`

値 YES の場合、uifont.ali ファイルの PDFEMBED パラメータで指定されたフォントが、PDF ドライバによって PDF 出力に組み込まれます。NO の場合、フォントは PDF 出力に追加されません。

デフォルト YES

A.4.75 PRINTJOB

表 A-78 では、PRINTJOB キーワードを使用できるコマンドを示します。

表 A-78 PRINTJOB キーワードを使用できるコマンド

rwclient	rwrn	rwbuilder	rwconverter	rwervlet	rwsgi	rwserver
×	×	○	×	×	×	×

説明 PRINTJOB を使用して、レポートを実行する前に「印刷ジョブ」ダイアログ・ボックスを表示させるかどうかを指定できます。

構文 PRINTJOB={YES|NO}

値 YES または NO

デフォルト NO

使用上の注意

- レポートが派生プロセスで実行されるときは（RWRUN などの実行ファイルが、RWBUILDER などの別の実行ファイル内で呼び出される場合）、PRINTJOB の値にかかわらず、「印刷ジョブ」ダイアログ・ボックスは表示されません。
- DESTYPE=MAIL のときは、PRINTJOB の値にかかわらず、「印刷ジョブ」ダイアログ・ボックスは表示されません。

A.4.76 READONLY

表 A-79 では、READONLY キーワードを使用できるコマンドを示します。

表 A-79 READONLY キーワードを使用できるコマンド

rwclient	rwrn	rwbuilder	rwconverter	rwervlet	rwsgi	rwserver
○	○	○	×	○	○	×

説明 READONLY を使用して、レポートでの複数回にわたる問合せで読取り一貫性を要求できます。Oracle からデータへのアクセス時に、SET TRANSACTION READ ONLY 文によって読取り一貫性を持たせることができます。

注意： SET TRANSACTION READ ONLY の詳細は、Oracle9i SQL のドキュメントを参照してください。

構文 READONLY={YES|NO}

値 YES は読取り一貫性を要求します。NO は読取り一貫性を要求しません。

デフォルト NO

使用上の注意

- このキーワードは、複数の問合せを使用するレポートでのみ使用します。単一の問合せを使用するレポートに対しては、読取り一貫性がロックなしで自動的に提供されます。
- Report トリガーの実行順では、データ・フェッチが発生する前に SET TRANSACTION READ ONLY を設定しておく必要があります。
- READONLY は、JSP として実行されるジョブでも使用できます。

A.4.77 REPLACEITEM

表 A-80 では、REPLACEITEM キーワードを使用できるコマンドを示します。

表 A-80 REPLACEITEM キーワードを使用できるコマンド

rwclient	rwrn	rwbuilder	rwconverter	rwservlet	rwcgi	rwserver
○	○	×	×	○	○	×

説明 REPLACEITEM を使用して、Oracle Portal に配信される現在のレポート出力による、同じ出力先に格納されている同じアイテムの旧バージョンの置換を指定できます。

構文 REPLACEITEM={YES|NO}

値 YES の場合、旧レポート出力は置換されます。NO の場合、旧バージョンは置換されません。この場合、新しい出力へのリンクが、同じレポートの旧バージョンへのリンクのリストに追加されます。

デフォルト なし

使用上の注意 このキーワードの使用はオプションです。関連するキーワードは、[CONTENTAREA](#)、[EXPIREDAYS](#)、[ITEMTITLE](#)、[OUTPUTFOLDER](#)、[OUTPUTPAGE](#)、[PAGEGROUP](#)、[REPLACEITEM](#)、[SCHEDULE](#)、[SITENAME](#)、[STATUSFOLDER](#) および [STATUSPAGE](#) です。

A.4.78 REPLYTO

表 A-81 では、REPLYTO キーワードを使用できるコマンドを示します。

表 A-81 REPLYTO キーワードを使用できるコマンド

rwclient	rwrn	rwbuilder	rwconverter	rwservlet	rwcgi	rwserver
○	○	×	×	○	○	×

説明 REPLYTO を使用して、送信者が送信者 ([FROM](#) キーワードで指定) 以外の宛先に返信を送信させる場合に、返信の送信先となる電子メール・アドレスを指定できます。

構文 REPLYTO=*someone@foo.com*

値 有効な電子メール・アドレス

デフォルト なし

使用上の注意 関連するキーワードは、[BCC](#)、[CC](#)、[FROM](#)、[REPLYTO](#) および [SUBJECT](#) です。電子メールの主な宛先を指定するには、[DESNAME](#) を使用します。

A.4.79 REPORT|MODULE

「[MODULE](#) | [REPORT](#)」を参照してください。

A.4.80 ROLE

表 A-82 では、ROLE キーワードを使用できるコマンドを示します。

表 A-82 ROLE キーワードを使用できるコマンド

rwclient	rwrn	rwbuilder	rwconverter	rwservlet	rwcgi	rwserver
○	○	×	×	○	○	×

説明 ROLE を使用して、レポートに対するデータベース・ロールを実行時にチェックするように指定できます。

- 構文

ROLE={rolename[/rolepassword]}
- 値

有効なロールと（オプションで）ロール・パスワード
- デフォルト

なし
- 使用上の注意

ROLE は、JSP として実行されるジョブでも使用できます。

A.4.81 RUNDEBUG

表 A-83 では、RUNDEBUG キーワードを使用できるコマンドを示します。

表 A-83 RUNDEBUG キーワードを使用できるコマンド

rwclient	rwrnrun	rwbuilder	rwconverter	rwervlet	rwcgi	rwserver
○	○	○	×	○	○	×

説明 RUNDEBUG を使用して、エラー・メッセージおよび警告をオンにできます。オンにしないとこれらは表示されません。たとえば、RUNDEBUG=YES と指定すると、次のようなエラー・メッセージが表示されます。「Frame 1 は Frame 2 と重複しますが、Frame 2 を含みません。この状況が許可されるかどうかは、実行されるジョブによって異なります。」

- 構文

RUNDEBUG={YES|NO}
- 値

YES の場合、追加のエラーおよび警告メッセージが表示されます。NO の場合、追加のエラーおよび警告メッセージが表示されません。
- デフォルト

YES
- 使用上の注意

RUNDEBUG は、JSP として実行されるジョブでも使用できます。

A.4.82 SAVE_RDF

表 A-84 では、SAVE_RDF キーワードを使用できるコマンドを示します。

表 A-84 SAVE_RDF キーワードを使用できるコマンド

rwclient	rwrnrun	rwbuilder	rwconverter	rwervlet	rwcgi	rwserver
×	○	○	×	×	×	×

説明 SAVE_RDF を使用して、複合 RDF ファイルおよび XML カスタマイズ・ファイルのファイル名を指定します。このキーワードは、CUSTOMIZE キーワードを使用して既存の RDF ファイルと Reports XML カスタマイズ・ファイルを複合し、その複合したものを新しい RDF ファイルに保存する場合に使用します。

構文 SAVE_RDF=*filename.rdf*

値 任意の有効なファイル名

デフォルト なし

使用上の注意 SAVE_RDF は、JSP ファイルにも使用できますが、ペーパー・レイアウト部分のみで、Web ソースには使用できません。

A.4.83 SCHEDULE

表 A-85 では、SCHEDULE キーワードを使用できるコマンドを示します。

表 A-85 SCHEDULE キーワードを使用できるコマンド

rwclient	rwrund	rwbuilder	rwconverter	rwservlet	rwcgi	rwserver
○	×	×	×	○	○	×

説明 SCHEDULE を使用して、レポートを実行する日付、時刻および頻度を設定できます。デフォルトは、1 回および即時のレポート実行です。時刻値は、24 時間方式で表します (例：1 時は 13 : 00)。スケジュール・コマンドを引用しなくてすむように、スペースのかわりにアンダースコア (_) を使用します。次に例を示します。

```
SCHEDULE=every_first_fri_of_month_from_15:53_Oct_23,_1999_retry_3_after_1_hour
SCHEDULE=last_weekday_before_15_from_15:53_Oct_23,_1999_retry_after_1_hour
```

または

```
SCHEDULE="every first fri of month from 15:53 Oct 23, 1999 retry 3 after 1 hour"
SCHEDULE="last weekday before 15 from 15:53 Oct 23, 1999 retry after 1 hour"
```

構文 SCHEDULE=*string*

ここで、*string* は次のように指定します。

```
[FREQ from] TIME [retry {n} + after LEN]
```

この文字列で使用される値は、表 A-86 で説明します。

表 A-86 SCHEDULE キーワードで利用できる文字列値

FREQ	hourly daily weekly monthly {every LEN DAYREPEAT}} {last {WEEKDAYS weekday weekend} before {n}+}
LEN	{n}+ {minute[s] hour[s] day[s] week[s] month[s]}
DAYREPEAT	{first second third fourth fifth} WEEKDAYS of month
WEEKDAYS	mon tue wed thu fri sat sun
TIME	now CLOCK [DATE]
CLOCK	h:m h:mm hh:m hh:mm
DATE	today tomorrow {MONTHS {d dd} [,year]}
MONTHS	jan feb mar apr may jun jul aug sep oct nov dec

デフォルト なし

A.4.84 SERVER

表 A-87 では、SERVER キーワードを使用できるコマンドを示します。

表 A-87 SERVER キーワードを使用できるコマンド

rwclient	rwrn	rwbuilder	rwconverter	rwservlet	rwcgi	rwserver
○	×	×	×	○	○	○

説明 SERVER を使用して、このレポートを実行する Reports Server の名前を指定できます。

構文 SERVER=servername

値 サーバー名、または TNS サービス・エントリ名（6i 互換サーバーを使用している場合）

デフォルト rwcgi の REPORTS_SERVER 環境変数で指定したサーバー名

使用上の注意 rwcgi で実行されるジョブについては、Web サーバー・マシン上で REPORTS_SERVER 環境変数を設定して、SERVER キーワードを省略すると、要求がデフォルト・サーバーを使用して処理されます。あるいは、SERVER キーワードを含めるとデフォルトが上書きされます。rwservlet で実行されるか JSP として実行されるジョブについては、デフォルト・サーバーをサーブレット構成ファイル（rwservlet.properties）で指定している場合は、SERVER キーワードを省略できます。あるいは、SERVER キーワードを含めると、デフォルトが上書きされます。

SERVER は、JSP として実行されるジョブでも使用できます。

A.4.85 SHOWENV

表 A-88 では、SHOWENV キーワードを使用できるコマンドを示します。

表 A-88 SHOWENV キーワードを使用できるコマンド

rwclient	rwrn	rwbuilder	rwconverter	rwervlet	rwcgi	rwserver
×	×	×	×	○	○	×

説明 SHOWENV を使用して、rwserver 構成ファイル（rwervlet.properties）を表示できます。

構文 `http://yourwebserver/reports/rwervlet/showenv[?]
[server=servername] [&authid=username/password]`

値 構文を参照してください。

デフォルト なし

A.4.86 SHOWJOBS

表 A-89 では、SHOWJOBS キーワードを使用できるコマンドを示します。

表 A-89 SHOWJOBS キーワードを使用できるコマンド

rwclient	rwrn	rwbuilder	rwconverter	rwervlet	rwcgi	rwserver
×	×	×	×	○	○	×

説明 SHOWJOBS を使用して、Reports Server のキュー・ステータスを Web 形式で表示できます。

構文 `http://yourwebserver/reports/rwervlet/showjobs[n] [?]
[server=server_name] [&authid=username/password] [&statusformat={html|xml|xmldtd}]`

値 構文を参照してください。

デフォルト なし

使用上の注意 Reports Server の名前が、環境変数またはサーブレット構成ファイルによって暗黙的に指定されているか、URL 要求で明示的に指定されている必要があります。リフレッシュ数 [n] はオプションです。この値を指定すると、レポートのキュー・ステータスが [n] 秒ごとに更新されます。STATUSFORMAT は、html、xml または xmldtd に設定できます。指定した形式でステータスが返されます。デフォルトは html です。

関連するキーワードは、[SERVER](#)、[AUTHID](#) および [STATUSFORMAT](#) です。

STATUSFORMAT パラメータは、`rwervlet` に対してのみ使用でき、`rwcgi` には使用できません。

A.4.87 SHOWMAP

[表 A-90](#) では、SHOWMAP キーワードを使用できるコマンドを示します。

表 A-90 SHOWMAP キーワードを使用できるコマンド

rwclient	rwrn	rwbuilder	rwconverter	rwervlet	rwcgi	rwserver
×	×	×	×	○	○	×

説明 SHOWMAP を使用して、`rwervlet` のキー・マッピングを表示できます。

構文 `http://yourwebserver/reports/rwervlet/showmap[?]
[server=servername] [&authid=username/password]`

値 構文を参照してください。

デフォルト なし

A.4.88 SHOWMYJOBS

[表 A-91](#) では、SHOWMYJOBS キーワードを使用できるコマンドを示します。

表 A-91 SHOWMYJOBS キーワードを使用できるコマンド

rwclient	rwrn	rwbuilder	rwconverter	rwervlet	rwcgi	rwserver
×	×	×	×	○	×	×

説明 SHOWMYJOBS を使用して、特定ユーザーに対する Reports Server のキュー・ステータスを表示できます。

構文 `http://yourwebserver/reports/rwervlet/showmyjobs[?]
[server=server_name] [&authid=username/password] [&statusformat={html|xml|xmldtd}]`

値 構文を参照してください。

デフォルト なし

使用上の注意 STATUSFORMAT は、html、xml または xmldtd に設定できます。指定した形式でステータスが返されます。デフォルトは html です。

関連するキーワードは、[SERVER](#)、[AUTHID](#) および [STATUSFORMAT](#) です。

STATUSFORMAT パラメータは、rwservlet に対してのみ使用でき、rwcgi には使用できません。

A.4.89 SHUTDOWN

[表 A-92](#) では、SHUTDOWN キーワードを使用できるコマンドを示します。

表 A-92 SHUTDOWN キーワードを使用できるコマンド

rwclient	rwrund	rwbuilder	rwconverter	rwservlet	rwcgi	rwserver
×	×	×	×	×	×	○

説明 SHUTDOWN を使用して、すでに実行中のサーバーを停止できます。また、[AUTHID](#) を使用してユーザー名とパスワードを指定する必要があります。

構文 SHUTDOWN={NORMAL|IMMEDIATE}

値 NORMAL または IMMEDIATE

デフォルト NORMAL

使用上の注意 SHUTDOWN キーワードを使用するユーザーは、Reports 管理ユーザーである必要があります。サーバーのセキュリティが有効な場合、サーバーは security API に問合せを行い、ユーザーのロールが停止を実行するのに適切かどうか（つまり、ユーザーが Reports 管理ユーザーかどうか）を判別します。セキュリティが有効でない場合も、ユーザーは、そのサーバーで定義された Reports 管理ユーザーである必要があります。

A.4.90 SITENAME

[表 A-93](#) では、SITENAME キーワードを使用できるコマンドを示します。

表 A-93 SITENAME キーワードを使用できるコマンド

rwclient	rwrund	rwbuilder	rwconverter	rwservlet	rwcgi	rwserver
○	○	×	×	○	○	×

説明 SITENAME を使用して、レポート出力の配信先となるサイトの名前を指定できます。Oracle Portal 3.0 ユーザーの場合、これはコンテンツ領域の名前になります（「[CONTENTAREA](#)」も参照してください）。このキーワードは、Oracle Portal の旧バージョン（および WebDB）に対する下位互換性を確保するために維持されています。Oracle9iAS Portal バージョン 2.0 以降では、[PAGEGROUP](#) を使用してください。

構文 SITENAME=*sitename*

値 Oracle Portal で使用されているサイト名

デフォルト なし

使用上の注意 レポート出力を Oracle Portal に配信する際、このキーワードの使用は必須です。値に文字スペース単位が含まれる場合や、cgicmd.dat ファイルで引数を指定する場合は、値の前後に引用符を付けます。

関連するキーワードは、[CONTENTAREA](#)、[EXPIREDAYS](#)、[ITEMTITLE](#)、[OUTPUTFOLDER](#)、[OUTPUTPAGE](#)、[PAGEGROUP](#)、[REPLACEITEM](#)、[SCHEDULE](#)、[SITENAME](#)、[STATUSFOLDER](#) および [STATUSPAGE](#) です。

ページやページ・グループなどの Oracle Portal オブジェクトには、表示名と内部名という 2 つの名前があります。Oracle Portal 出力で使用するオブジェクトを Oracle Portal 内で作成する際、Oracle Portal のオンライン・ヘルプで指定された内部名の命名規則に従って、内部名と表示名を同じにします。これにより、Reports コマンドラインで Portal 関連のキーワードに値を指定する際、どちらの名前を指定すればよいか考慮する必要がなくなります。

A.4.91 SOURCE

表 A-94 では、SOURCE キーワードを使用できるコマンドを示します。

表 A-94 SOURCE キーワードを使用できるコマンド

rwclient	rwrn	rwbuilder	rwconverter	rwservlet	rwcgi	rwserver
×	×	×	○	×	×	×

説明 SOURCE を使用して、変換する 1 つのレポートまたはライブラリ、あるいはそれらのリストを指定できます。rwconverter コマンドを実行するには、ソース・レポートまたはソース・ライブラリを指定する必要があります。

構文 SOURCE={*source_name* | (*source_name1*, *source_name2*, ...)}

値 有効なレポート名、ライブラリ名またはファイル名、あるいは名前をカンマで区切り、括弧で囲んだそれらのリスト ((*qanda*, *test*, *dmast*) など) を指定します。

デフォルト なし

使用上の注意

- データベースに格納されているレポートまたはライブラリには、SQL のワイルドカード文字（% および _）を使用できます。たとえば R% を指定すると、データベースに格納されているレポートのうち R で始まるものがすべてフェッチされます。該当するすべてのレポートが変換されます。
- レポート名、ライブラリ名、ファイル名のリストは、名前をカンマで区切り、括弧で囲む必要があります。次に例を示します。

(qanda,test,dmast) OR (qanda, test, dmast)
- ワイルドカード文字は、ファイルに格納されているレポートまたはライブラリ（拡張子 rdf、rep、rex、pld、pll または xml を持つもの）には無効です。
- SOURCE キーワードの値は、オペレーティング・システムによって異なります。
- ユーザー所有の Reports Builder 表を使用する場合、複数のユーザーからのレポートまたはライブラリは、各ユーザー用に個別に変換する必要があります。
- レポートまたはライブラリを変換するには、これらを自分で作成済みであるか、または自分が作成したものでない場合は、アクセス権限が付与されている必要があります。レポートまたはライブラリ名の前にユーザー ID が付いていない場合、ユーザー ID は現行のユーザーになります。
- DTYPE=REGISTER である場合、定義タイプやフォーマット、ユーザー・アクセス、および使用可能日カレンダーなど、共通のパラメータを持つレポート定義ファイルのみ指定しなければならない場合があります。

A.4.92 SSOCONN

表 A-95 では、SSOCONN キーワードを使用できるコマンドを示します。

表 A-95 SSOCONN キーワードを使用できるコマンド

rwclient	rwrn	rwbuilder	rwconverter	rwservlet	rwcgi	rwserver
×	×	×	×	○	×	×

説明 SSOCONN を使用して、シングル・サインオン環境で 1 つ以上のデータ・ソースへの接続に使用する 1 つ以上の接続文字列を指定できます。

構文 SSOCONN=key[/type[/conn_str]][,key[/type[/conn_str]]]

次に例を示します。

ssoconn=mykey/OracleDB/userid

値 SSOCONN 構文で表される変数値に関する情報を次に示します。

- *key* は、Oracle Internet Directory (OID) に格納される接続文字列値を参照します。
- *type* は、レポートのデータ・ソース・タイプを指定する事前定義の文字列です。Oracle9iAS のインストール時に提供されるタイプには、OracleDB、JDBCPDS (Java Database Connectivity Pluggable Data Source) および ExpressPDS (Oracle Express Pluggable Data Source) があります。
- *conn_str* は、レポート実行のために *rwervlet* によって構成される接続文字列パラメータの名前です。

デフォルト なし

使用上の注意

- レポートに複数のデータ・ソースが使用されている場合、データ・ソース接続文字列をカンマで区切ります。次に例を示します。
`ssoconn=key1/type1/conn_str,key2/type2/conn_str2,key3/type3/conn_str3`
- SSOCONN 引数の簡易バージョンもあります。表 A-96 に例を示します。

表 A-96 SSOCONN 引数の簡易バージョン

引数	説明
ssoconn=mkey	キー名のみ指定すると、デフォルトのタイプ (Oracle DB) が使用され、接続パラメータは USERID キーワードになります。
ssoconn=mkey/PDSApp	キー名とアプリケーション・タイプの両方を指定すると、接続パラメータは USERID キーワードになります。
ssoconn=mkey/PDSApp/P_1	すべてを指定すると、指定された値が使用されます。

- SSOCONN は、JSP として実行されるジョブでも使用できます。

注意： Reports とシングル・サインオン (SSO) の詳細は、第7章「データ・ソース・シングル・サインオン」を参照してください。

A.4.93 STATUSFORMAT

表 A-97 では、STATUSFORMAT キーワードを使用できるコマンドを示します。

表 A-97 STATUSFORMAT キーワードを使用できるコマンド

rwclient	rwrn	rwbuilder	rwconverter	rwervlet	rwcgi	rwserver
×	×	×	×	○	×	×

説明 STATUSFORMAT を使用して、Reports Server のキュー・ステータスを Web で表示する際の形式を指定できます。

構文 Reports_URL/rwervlet/showjobs?
server=server_name&statusformat={html|xml|xmldtd}

値 HTML、XML または XMLDTD を指定します。HTML を使用すると、レポートのキュー・ステータスが HTML 形式で出力されるよう指定できます。XML を使用すると、XML 形式で出力されるよう指定できます。XMLDTD を使用すると、インライン・データ型定義情報を持つ XML 形式で出力されるよう指定できます。

デフォルト HTML

使用上の注意 STATUSFORMAT は、SHOWJOBS および SHOWMYJOBS キーワードとともに使用します。

A.4.94 STATUSFOLDER

表 A-98 では、STATUSFOLDER キーワードを使用できるコマンドを示します。

表 A-98 STATUSFOLDER キーワードを使用できるコマンド

rwclient	rwrn	rwbuilder	rwconverter	rwervlet	rwcgi	rwserver
○	○	×	×	○	○	×

説明 STATUSFOLDER を使用して、ステータス情報の配信先となるフォルダを指定できます。この指定を省略すると、"Oracle_Reports_Status" という新規フォルダが作成されます。この値は、Oracle Portal の旧バージョンに対する下位互換性を確保するために維持されています。現行バージョン (Oracle9iAS Portal バージョン 2.0 以降) の場合は、「STATUSPAGE」を参照してください。

構文 STATUSFOLDER=Oracle_Reports_Status

値 Oracle Portal で使用されているフォルダ名

デフォルト Oracle_Reports_Status

使用上の注意 このキーワードの値は大文字と小文字を区別します。このキーワードの使用はオプションです。

値に文字スペース単位が含まれる場合や、cgicmd.dat ファイルで引数を指定する場合は、値の前後に引用符を付けます。

ページやページ・グループなどの Oracle Portal オブジェクトには、表示名と内部名という 2 つの名前があります。Oracle Portal 出力で使用するオブジェクトを Oracle Portal 内で作成する際、Oracle Portal のオンライン・ヘルプで指定された内部名の命名規則に従って、内部名と表示名を同じにします。これにより、Reports コマンドラインで Portal 関連のキーワードに値を指定する際、どちらの名前を指定すればよいか考慮する必要がなくなります。

A.4.95 STATUSPAGE

表 A-99 では、STATUSPAGE キーワードを使用できるコマンドを示します。

表 A-99 STATUSPAGE キーワードを使用できるコマンド

rwclient	rwrwn	rwbuilder	rwconverter	rwervlet	rwcgi	rwserver
○	○	×	×	○	○	×

説明 STATUSPAGE を使用して、ジョブ・ステータス情報の配信先となるページを指定できます。この指定を省略すると、"Oracle_Reports_Status" という新規ページが作成されます。このキーワードは、Oracle9iAS Portal バージョン 2.0 以降に対して使用してください。下位互換性については、「STATUSFOLDER」を参照してください。

構文 STATUSPAGE=Oracle_Reports_Status

値 Oracle Portal で使用されているページ名

デフォルト Oracle_Reports_Status

使用上の注意 このキーワードの値は大文字と小文字を区別します。このキーワードの使用はオプションです。

値に文字スペース単位が含まれる場合や、cgicmd.dat ファイルで引数を指定する場合は、値の前後に引用符を付けます。

ページやページ・グループなどの Oracle Portal オブジェクトには、表示名と内部名という 2 つの名前があります。Oracle Portal 出力で使用するオブジェクトを Oracle Portal 内で作成する際、Oracle Portal のオンライン・ヘルプで指定された内部名の命名規則に従って、内部名と表示名を同じにします。これにより、Reports コマンドラインで Portal 関連のキーワードに値を指定する際、どちらの名前を指定すればよいか考慮する必要がなくなります。

A.4.96 STYPE

表 A-100 では、STYPE キーワードを使用できるコマンドを示します。

表 A-100 STYPE キーワードを使用できるコマンド

rwclient	rwrund	rwbuilder	rwconverter	rservlet	rwcgi	rwserver
×	×	×	○	×	×	×

説明 STYPE を使用して、変換するレポートまたはライブラリの形式を指定できます。

構文 STYPE={PLDFILE|PLLFILE|RDFFILE|REXFILE|XMLFILE|JSPFILE}

値 次の値をどれか1つ指定します。

- PLDFILE の場合、ソース PL/SQL ライブラリは ASCII 形式のファイルに格納されています。
- PLLFILE の場合、ソース PL/SQL ライブラリは、ソース・コードおよび P コード（コンパイル済 PL/SQL）を含むファイルに格納されています。
- RDFFILE の場合、ソース・レポートは1つ以上のレポート定義ファイル（拡張子 `rdf` を持つファイル）に格納されています。
- REXFILE の場合、ソース・レポートは1つ以上のテキスト・ファイル（拡張子 `rex` を持つファイル）に格納されています。
- XMLFILE の場合、ソース・レポートは1つ以上の XML ファイルに格納されています。
- JSPFILE の場合、ソース・レポートは1つ以上の JSP ファイルに格納されています。

デフォルト RDFFILE

使用上の注意 DTYPE=REGISTER である場合、STYPE には RDFFILE、REXFILE、XML または JSPFILE を選択します。

A.4.97 SUBJECT

表 A-101 では、SUBJECT キーワードを使用できるコマンドを示します。

表 A-101 SUBJECT キーワードを使用できるコマンド

rwclient	rwrund	rwbuilder	rwconverter	rservlet	rwcgi	rwserver
○	○	×	×	○	○	×

説明 SUBJECT を使用して、電子メールの件名を指定できます。

構文 SUBJECT="any text"

値 任意のテキスト

デフォルト なし

使用上の注意 文字スペースを含む件名は、引用符で囲みます。単一語句の件名には引用符は不要です。

関連するキーワードは、[BCC](#)、[CC](#)、[FROM](#)、[REPLYTO](#) および [SUBJECT](#) です。電子メールの主な宛先を指定するには、[DESNAME](#) を使用します。

A.4.98 TOLERANCE

[表 A-102](#) では、TOLERANCE キーワードを使用できるコマンドを示します。

表 A-102 TOLERANCE キーワードを使用できるコマンド

rwclient	rwrund	rwbuilder	rwconverter	rwservlet	rwcgi	rwserver
○	×	×	×	○	○	×

説明 TOLERANCE を使用して、重複したジョブが検出された場合の、レポートのキャッシュ出力の再利用に対する最大許容時間（分）を設定できます。レポートについて許容時間を設定すると、重複したジョブが見つかった場合の処理時間が短縮されます。

重複ジョブの検出の詳細は、[8.8 項「キャッシュからのレポート出力の再使用」](#)（第 8 章）を参照してください。

構文 TOLERANCE=time_string

値 時刻文字列には、次のいずれかの形式を使用できます。

- *n{unit}* は、オプションの単位を持つ数値の場合に使用します。**unit** には、分 (s)、時間 (s) または日 (s) を指定できます。単位を指定しない場合は、デフォルトの分 (s) が使用されます。
- {*Mon DD, YYYY*} *hh:mi:ss* am|pm {*timezone*} は、日付 / 時刻形式の場合に使用します。日付情報はオプションです。日付を指定しない場合は、本日の日付が使用されます。タイムゾーンもオプションです。タイムゾーンを指定しない場合は、**Reports Server** のタイムゾーンが使用されます。日付 / 時刻には必ず米国のロケールを使用します。この形式は、Java の **DateFormat.MEDIUM** タイプの定義と同じです。

デフォルト なし

使用上の注意

- TOLERANCE が設定されていないときは、キャッシュに重複レポートが見つかった場合でも、Oracle9iAS Reports Services はそのレポートを再実行します。
- 同一ジョブが送信されたときにレポートが処理中（つまり、現行ジョブ・キューにあるとき）であった場合は、TOLERANCE が設定されていない場合やゼロに設定されている場合でも、Oracle9iAS Reports Services は現在実行中のジョブの出力を再利用します。

A.4.99 TRACEFILE

表 A-103 では、TRACEFILE キーワードを使用できるコマンドを示します。

表 A-103 TRACEFILE キーワードを使用できるコマンド

rwclient	rwrund	rwbuilder	rwconverter	rwervlet	rwcgi	rwserver
×	○	○	×	×	×	×

説明 TRACEFILE は、トレース情報を書き込むファイルの名前です。

注意： ランタイム環境で、または Reports Services コンポーネントを監視するとき、3つのすべての TRACE キーワードを使用できます。ただしサーバー・セキュリティの場合、サーバー環境では TRACEOPTS のみが使用できます。これにより、ファイルが Reports Server のファイル・システムに任意に書き込まれることを防止できます。

Reports Server のトレースは、サーバー構成ファイル（<server_name>.conf）で構成します（詳細は、第3章を参照してください）。Reports Servlet のトレースは、サーブレット構成ファイル（rwservlet.properties）で構成します（詳細は、第3章を参照してください）。個別ジョブのトレースは、ランタイム・コマンドラインで TRACEOPTS コマンドライン引数を使用して指定します。

構文 TRACEFILE=tracefile

値 任意の有効なファイル名

デフォルト なし

使用上の注意 Reports でのトレース実行方法の詳細は、14.5.1.1 項「Trace の概要」を参照してください。

A.4.100 TRACEMODE

表 A-104 では、TRACEMODE キーワードを使用できるコマンドを示します。

表 A-104 TRACEMODE キーワードを使用できるコマンド

rwclient	rwrund	rwbuilder	rwconverter	rwervlet	rwcgi	rwserver
○	○	○	×	○	○	×

説明 TRACEMODE は、新規トレース情報をトレース・ファイルの既存情報に追加するか、またはファイル全体を上書きするかを指定します。

構文 TRACEMODE={TRACE_APPEND|TRACE_REPLACE}

値 TRACE_APPEND は、新しい情報をファイルの末尾に追加します。TRACE_REPLACE はファイルを上書きします。

デフォルト TRACE_APPEND

使用上の注意 Reports でのトレース実行方法の詳細は、14.5.1.1 項「Trace の概要」を参照してください。

A.4.101 TRACEOPTS

表 A-105 では、TRACEOPTS キーワードを使用できるコマンドを示します。

表 A-105 TRACEOPTS キーワードを使用できるコマンド

rwclient	rwrund	rwbuilder	rwconverter	rwervlet	rwcgi	rwserver
○	○	○	×	○	○	○

説明 TRACEOPTS は、レポート実行中にトレース・ファイルに記録するトレース情報を指示します。

構文 TRACEOPTS={TRACE_ERR|TRACE_PRF|TRACE_APP|TRACE_PLS|TRACE_SQL|TRACE_TMS|TRACE_DST|TRACE_ALL|TRACE_EXC|(TRACE_ERR, TRACE_PLS, ...)}

値 次の値が適用されます。

- 括弧に入ったオプションのリストは、括弧内のすべてのオプションを使用することを指定します。たとえば、TRACE_OPTS=(TRACE_APP, TRACE_PRF) は、TRACE_APP および TRACE_PRF を適用するということです。
- TRACE_ALL の場合、トレース・ファイルに、考えられるすべてのトレース情報のログをとります。
- TRACE_APP の場合、トレース・ファイルに、すべてのレポート・オブジェクトのトレース情報のログをとります。
- TRACE_BRK の場合、トレース・ファイルに、すべてのブレーク・ポイントを示します。
- TRACE_DBG の場合、デバッグ情報のログをとります。
- TRACE_DST の場合、トレース・ファイルに宛先リストを示します。この情報を使用して、どのセクションがどの宛先に送られたかを調べることができます。
- TRACE_ERR の場合、トレース・ファイルにエラー・メッセージと警告を示します。
- TRACE_EXC の場合、Reports Server 例外を示します。
- TRACE_INF の場合、他のオプションでトレース・ファイルに記録されない情報をすべて記録します。
- TRACE_LOG の場合、トレース・ファイルのログ情報を複製します。サーバー構成ファイルで *trace* 要素の他に *log* 要素も指定した場合、ログ・ファイルに出力された情報がトレース・ファイルにも送信されます。
- TRACE_PLS の場合、トレース・ファイルに、すべての PL/SQL オブジェクトのトレース情報のログをとります。
- TRACE_PRF の場合、トレース・ファイルに、パフォーマンス統計のログをとります。
- TRACE_SQL の場合、トレース・ファイルに、すべての SQL に関するトレース情報のログをとります。
- TRACE_STA は、初期化中、準備完了、実行中および停止など、サーバーおよびエンジンの状態に関する情報を提供します。
- TRACE_TMS は、トレース・ファイルの各エントリに対してタイムスタンプを入力します。
- TRACE_WRN は、サーバーの警告メッセージを示します。

デフォルト TRACE_ALL

使用上の注意 TRACEOPTS は、JSP として実行されるジョブでも使用できます。

Reports でのトレース実行方法の詳細は、[14.5.1.1 項「Trace の概要」](#)を参照してください。

A.4.102 UNINSTALL

表 A-106 では、UNINSTALL キーワードを使用できるコマンドを示します。

表 A-106 UNINSTALL キーワードを使用できるコマンド

rwclient	rwrund	rwbuilder	rwconverter	rwservlet	rwcgi	rwserver
×	×	×	×	×	×	○

説明 UNINSTALL を使用して、Microsoft Windows から Reports Server インスタンスを削除できます。このプロセスは、システムから Reports Server サービスを削除します。このキーワードは、UNIX では無視されます。

構文 -UNINSTALL REPORTS_SERVER_NAME

値 既存の Reports Server インスタンス名

デフォルト なし

使用上の注意 INSTALL とともに BATCH=YES を使用すると、通常は削除プロセス時に表示されるプロンプトおよびダイアログが表示されなくなります。

A.4.103 URLPARAMETER

表 A-107 では、URLPARAMETER キーワードを使用できるコマンドを示します。

表 A-107 URLPARAMETER キーワードを使用できるコマンド

rwclient	rwrund	rwbuilder	rwconverter	rwservlet	rwcgi	rwserver
○	×	×	×	○	○	×

説明 URLPARAMETER を使用して、URL エンジンでフェッチする URL を指定できます。

構文 URLPARAMETER=http://webserver_name/pagename.html

値 有効な URL

デフォルト なし

使用上の注意 このキーワードは、Reports Server 構成ファイルで *job* 要素の *jobType* パラメータに *rwurl* が指定されており、URL エンジンが存在する場合に使用できます。

A.4.104 USERID

表 A-108 では、USERID キーワードを使用できるコマンドを示します。

表 A-108 USERID キーワードを使用できるコマンド

rwclient	rwrund	rwbuilder	rwconverter	rwervlet	rwcgi	rwserver
○	○	○	○	○	○	×

説明 USERID は、シングル・サインオンを使用しない場合にのみ使用します。USERID を使用して、Oracle ユーザー名およびパスワードを、リモート・データベースにアクセスするためのオプションのデータベース名とともに指定できます。パスワードを省略すると、ユーザーにレポートの実行が許可される前に、データベース・ログオン・フォームが自動的に表示されます。

ユーザーのデータベースへのログオンを許可するには、レポート要求から USERID キーワードのパスワード部分を省いてください。レポート要求を実行するたびにユーザーにログオンさせるようにするには、レポートのキー・マップ・ファイル `cgicmd.dat` を使用してランタイム・コマンドを指定し、%D 引数を関連のキー・マップ・エントリに含めます。

注意： `cgicmd.dat` ファイルの使用の詳細は、[第 8 章「レポート要求の実行」](#)を参照してください。

構文 `userid=username[/password] [@database]`

値 ログイン定義は次の形式のいずれかで、長さは 512 バイトを超えてはいけません。

`username[/password]`
`username[/password] [@database]`

デフォルト なし

使用上の注意 USERID は、JSP として実行されるジョブでも使用できます。

A.4.105 WEBSERVER_DEBUG

表 A-109 では、WEBSERVER_DEBUG キーワードを使用できるコマンドを示します。

表 A-109 WEBSERVER_DEBUG キーワードを使用できるコマンド

rwclient	rwrund	rwbuilder	rwconverter	rwervlet	rwcgi	rwserver
×	×	○	×	×	×	×

説明 WEBSERVER_DEBUG を使用して、JSP のデバッグを実行できます。このキーワードを使用すると、stderr.log および stdout.log ファイルが docroot/port#ディレクトリに作成され、JSP 一時ファイルは docroot/port#/default に、ログ・ファイルは docroot/port#/log に残されるので、ユーザーが調べることができます。

構文 WEBSERVER_DEBUG={YES|NO}

値 YES の場合、デバッグ・ファイルが作成されます。NO の場合、デバッグ・ファイルは作成されません。

デフォルト NO

使用上の注意 このキーワードは、ジョブを JSP として実行している場合のみ使用してください。関連するキーワードは、WEBSERVER_DEBUG、WEBSERVER_DOCROOT、WEBSERVER_PORT です。

A.4.106 WEBSERVER_DOCROOT

表 A-110 では、WEBSERVER_DOCROOT キーワードを使用できるコマンドを示します。

表 A-110 WEBSERVER_DOCROOT キーワードを使用できるコマンド

rwclient	rwruntime	rwbuilder	rwconverter	rwservlet	rwsgi	rwserver
×	×	○	×	×	×	×

説明 WEBSERVER_DOCROOT を使用して、Reports Builder ドキュメント・ルート・ディレクトリを設定できます。イメージ、HTML など JSP で参照するすべてのファイルは、このディレクトリに対する相対位置で指定される必要があります。ドキュメント・ルートを作業ディレクトリに設定することで、これらのファイルをコピーする必要がなくなります。

構文 WEBSERVER_DOCROOT=REPORTS_TMP/docroot

次に例を示します。

WEBSERVER_DOCROOT=c:¥temp¥docroot

値 Reports 一時フォルダ内のドキュメント・ルート・フォルダへのディレクトリ

デフォルト なし

使用上の注意 このキーワードは、ジョブを JSP として実行している場合のみ使用してください。関連するキーワードは、WEBSERVER_DEBUG、WEBSERVER_DOCROOT、WEBSERVER_PORT です。

A.4.107 WEBSERVER_PORT

表 A-111 では、WEBSERVER_PORT キーワードを使用できるコマンドを示します。

表 A-111 WEBSERVER_PORT キーワードを使用できるコマンド

rwclient	rwrund	rwbuilder	rwconverter	rwervlet	rwcgi	rwserver
×	×	○	×	×	×	×

説明 WEBSERVER_PORT を使用して、内部 Web サーバーがリスニングするポート番号を指定できます。単一のポート番号（3002 など）またはポート番号の範囲（3100-3200 など）を指定できます。単一のポート番号を指定した場合、Reports は内部 Web サーバーのリスニングをそのポートから開始します。そのポートが使用中の場合、次に利用可能なポートに移ります。ポート番号の範囲を指定した場合、Reports はその範囲内で空きポートを検索します。

構文 WEBSERVER_PORT=*port number or range of numbers*

値 有効なポート番号またはポート番号の範囲

デフォルト デフォルトのポートは 3000 です。デフォルトのポート範囲は 3000-3010 です。

使用上の注意 このキーワードは、ジョブを JSP として実行している場合のみ使用してください。関連するキーワードは、WEBSERVER_DEBUG、WEBSERVER_DOCROOT、WEBSERVER_PORT です。

Reports 関連の環境変数

環境変数とは、Oracle9iAS Reports Services をホスティングする環境を構成するパラメータです。関連する環境変数には、Oracle9iAS のインストーラによって自動的にデフォルト値が定義されます。デフォルトの環境を一部変更する場合は、環境変数の設定を編集できます。

- Windows NT の場合は、レジストリ・エディタで環境変数を編集します（「スタート」⇒「ファイル名を指定して実行」⇒「Regedit」）。
- UNIX の場合は、初期のデフォルト値を定義するシェル・スクリプトを修正して実行することで、環境変数を編集します。この場合、必ず変更前の `reports.sh` ファイルのバックアップを作成しておいてください。

表 B-1 では、Oracle9iAS Reports Services に関連する環境変数を説明します。

注意： NLS のすべての環境変数の詳細は、『Oracle9i Database グローバリゼーション・サポート・ガイド』を参照してください。

表 B-1 Oracle9iAS Reports Services に関連する環境変数

変数	説明
DEVELOPER_NLS_LANG	Oracle9iDS Reports Builder の言語。この環境変数とその有効値の詳細は、 第 12 章「NLS と双方向サポート」 を参照してください。
NLS_CALENDAR	使用するカレンダー・システム。
NLS_CREDIT	正の通貨値を示す文字列。
NLS_CURRENCY	各国通貨記号。
NLS_DATE_FORMAT	日付に使用されるデフォルトの書式マスク。
NLS_DATE_LANGUAGE	日付に使用されるデフォルト言語。
NLS_DEBIT	負の通貨値を示す文字列。
NLS_ISO_CURRENCY	ISO 通貨記号。
NLS_LANG	Oracle9iAS Reports Services が使用する言語設定。この環境変数とその有効値の詳細は、 第 12 章「NLS と双方向サポート」 を参照してください。
NLS_LIST_SEPARATOR	リスト内の項目を区切るための文字。
NLS_MONETARY_CHARACTERS	通貨値用の小数点文字と 3 桁区切り。
NLS_NUMERIC_CHARACTERS	数値用の小数点文字とグループ・セパレータ。
NLS_SORT	文字データに使用されるソートのタイプ。
REPORTS_COOKIE_EXPIRE	(rwcgi のみ) Cookie の期限を分単位で決定します。デフォルト値は 30 です。 Cookie は、保護された Oracle9iAS Reports Server にユーザーがログインしてレポート要求を実行する際に、暗号化されたユーザー名とパスワードをクライアント側に保存します。ユーザーがログインに成功すると、ユーザーのブラウザに暗号化された Cookie が送られます。Cookie が期限切れになった場合、ユーザーはその後の要求を実行するために再度認証を受ける必要があります。
REPORTS_DB_AUTH	(rwcgi のみ) データベースにログインするためのデータベース認証テンプレートを指定します。デフォルト値は dbauth.htm です。
REPORTS_ENCRYPTION_KEY	(rwcgi のみ) Cookie 用にユーザー名とパスワードを暗号化する際に使用される暗号化キーを指定します。暗号化キーには任意の文字列を設定できます。デフォルト値は reports9.0 です。

表 B-1 Oracle9iAS Reports Services に関連する環境変数（続き）

変数	説明
REPORTS_CGIDIAGBODYTAGS	<p>(<i>rwcgi</i> のみ) Oracle9iAS Reports CGI に使用します。RWCGI の診断およびデバッグ出力に <BODY...> タグとして挿入される HTML タグを指定します。たとえば、この環境変数を使用して、テキストと背景色または背景イメージを設定できます。</p> <p>Reports CGI は、下位互換性を確保するためにのみ維持されています。</p>
REPORTS_CGIDIAGHEADTAGS	<p>(<i>rwcgi</i> のみ) Oracle9iAS Reports CGI に使用します。RWCGI の診断およびデバッグ出力の中で、<HEAD>...</HEAD> タグの間に挿入される HTML タグを指定します。たとえば、この環境変数を使用して <TITLE> または <META> タグを設定できます。</p> <p>Reports CGI は、下位互換性を確保するためにのみ維持されています。</p>
REPORTS_CGIHELP	<p>(<i>rwcgi</i> のみ) Oracle9iAS Reports CGI に使用します。RWCGI が次に示す空の要求で起動された場合に表示される RWCGI ヘルプ・ファイルの URL および URI を定義します。</p> <p><code>http://<your_webserver>/RWCGI90?</code></p> <p>たとえば、これを <code>http://www.yahoo.com</code> に設定するとその URL に移動し、<code>myhelpfile.htm</code> に設定すると次のファイルが表示されます。</p> <p><code>http://<your_webserver>/myhelpfile.htm</code></p> <p>このパラメータが未定義の場合、デフォルトのヘルプ画面が表示されます。</p> <p>Reports CGI は、下位互換性を確保するためにのみ維持されています。</p>
REPORTS_CGIMAP	<p>(<i>rwcgi</i> のみ) Oracle9iAS Reports CGI に使用します。マップ・ファイル構成を使用する場合に、RWCGI マップ・ファイルの完全修飾ファイル名および位置を定義します。次に例を示します。</p> <p><code>ORACLE_HOME\reports\conf\cgicmd.dat</code></p> <p>Reports CGI は、下位互換性を確保するためにのみ維持されています。</p>

表 B-1 Oracle9iAS Reports Services に関連する環境変数（続き）

変数	説明
REPORTS_CGINODIAG	<p>(<code>rwcgi</code> のみ) YES または NO に設定します。 Oracle9iAS Reports CGI に使用します。これを定義すると、ヘルプやショーマップといった、<code>RWCGI</code> からのデバッグおよび診断の出力がすべて無効になります。</p> <p>たとえば、<code>REPORTS_CGINODIAG</code> が定義されていると、次の要求が機能しなくなります。</p> <p><code>http://<your_webserver>/rwcgi/help?</code></p> <p>Reports CGI は、下位互換性を確保するためにのみ維持されています。</p>
REPORTS_PATH	<p>Reports のコンポーネントが必要なファイルを自動的に検索するディレクトリを指定します。複数のディレクトリは、プラットフォームに応じてセミコロンまたはコロンで区切ります。</p> <p>Reports Server 構成ファイル (<code><server_name>.conf</code>) に含まれる <i>engine</i> 要素の <i>sourceDir</i> 属性にパスを指定すると、その値でここに設定した値が上書きされます。</p>
REPORTS_SERVER	<p>(<code>rwcgi</code> のみ) デフォルトの Oracle9iAS Reports Server を指定します。</p> <p>この環境変数が設定されているときは、デフォルト・サーバーでレポート要求を処理する場合に、その要求で <code>SERVER</code> コマンドライン引数を省略できます。反対に、<code>SERVER</code> コマンドライン引数を指定することで、この環境変数に指定したデフォルト値を無効にできます。</p>
REPORTS_SSLPORT	<p>(<code>rwcgi</code> のみ) SSL を使用する場合に 443 以外のポート番号を使うときは、この変数を使用して別のポート番号を指定できます。デフォルト値は 443 です。</p>
REPORTS_SYS_AUTH	<p>(<code>rwcgi</code> のみ) アクセスが制限されている Oracle9iAS Reports Server に対してユーザーがレポート要求を実行するときに、ユーザー名とパスワードを要求するダイアログを表示するための認証テンプレートを指定します。</p>
REPORTS_TMP	<p>Reports の開発関連およびサーバー関連一時ファイルを自動的に（かつ一時的）に保存する一時ディレクトリを指定します。</p>
RW	<p>Oracle ホームにおけるレポート固有のディレクトリを指定します。次に例を示します。</p> <p>Windows: <code>RW=d:\¥%ORACLE_HOME%\reports</code></p> <p>UNIX: <code>RW=\$ORACLE_HOME/reports</code></p>

表 B-1 Oracle9iAS Reports Services に関連する環境変数（続き）

変数	説明
USER_NLS_LANG	Oracle9iAS Reports Runtime コンポーネントの言語。この環境変数とその有効値の詳細は、 第 12 章「NLS と双方向サポート」 を参照してください。

Oracle9iAS Portal へのレポートのバッチ登録

Oracle9iAS Portal に多数のレポートを登録する場合は、Oracle9iAS Portal ユーザー・インタフェースで個別に登録するのではなく、バッチ・スクリプトを使用してまとめて登録できます。同様に、多数のレポートを登録解除する場合にも、バッチ・スクリプトを使用すれば効率的です。

- [レポート定義ファイルのバッチ登録](#)
- [レポート・パッケージのバッチ削除](#)
- [PL/SQL バッチ登録ファンクション](#)

C.1 レポート定義ファイルのバッチ登録

Oracle9iAS Portal でレポートをバッチ登録するには、次の手順を実行します。

1. [RWCONVERTER](#) の実行による [SQL スクリプトの生成](#)
2. [SQL*Plus](#) でのスクリプトの実行

C.1.1 RWCONVERTER の実行による SQL スクリプトの生成

SQL*Plus で実行可能なレポート登録用 SQL スクリプトを生成する手順は、次のとおりです。

1. オペレーティング・システム・プロンプト（DOS または UNIX）から、RWCONVERTER コマンドをキーワードとともに入力して、レポート定義ファイルをバッチ登録します。RWCONVERTER キーワードの詳細は、[付録 A「コマンドライン回数」](#)を参照してください。

注意： 必要なロード関数を持つスクリプト・ファイルを作成するには、DTYPE、STYPE、SOURCE および DEST 引数を指定します。Oracle9iAS Portal で機能パッケージを作成するには、スクリプト・ファイルの作成に使用した引数に加え、P_SERVERS、P_PRIVILEGE、P_TYPES、P_FORMATS を指定する必要があります。

次に、Microsoft Windows での RWCONVERTER コマンドラインの例を示します。

```
rwconverter.exe dtype="register" stype="rdf" file="
source="(security.rdf,earnings.rdf,acc_pay.rdf)" dest="(output.sql)"
p_owner="Oracle9iAS Portal" p_servers="(repserver,acct_server)"
p_description="restricted report" p_privilege="(SCOTT,JABERS,ACCT)"
p_availability="production" p_types="(Cache,printer)"
p_formats="(HTMLCSS,PDF)" p_printers="(sales_printer,acct_printer)"
p_pformTemplate="public.finance_template"
p_trigger="Is begin IF UPPER(DESTTYPE) = 'PRINTER' AND
EMPNAME = 'SMITH' THEN RETURN(TRUE); ELSE RETURN(FALSE); END IF; end;"
```

前述のコマンドラインを指定すると、output.sql という名前で、次の内容を含む SQL スクリプト・ファイルが生成されます。

```
SET SERVEROUTPUT ON
```

```
VAR STATUS NUMBER;
```

```
EXEC :STATUS := RWWWVREG.REGISTER_REPORT (P_NAME=>'Security',
P_OWNER=>'Oracle9iAS Portal', P_SERVERS=>'repserver,acct_server',
P_FILENAME=>'security.rdf', P_DESCRIPTION=>'restricted report',
P_PRIVILEGE=>'SCOTT,JABERS,ACCT', P_AVAILABILITY=>'production'
```

```

P_TYPES=>'Cache,printer', P_FORMATS=>'HTMLCSS,PDF)',
P_PRINTERS=>'sales_printer,acct_printer
P_PFORMTEMPLATE=>'public.finance_template' P_PARAMETERS=>'(P_LASTNAME)
(P_SSN)', P_TRIGGER=>'Is begin IF UPPER(DESTYPE) = 'PRINTER' AND
EMPNAME = 'SMITH' THEN RETURN(TRUE); ELSE RETURN(FALSE); END IF; end;');

EXEC :STATUS := RWWWVREG.REGISTER_REPORT (P_NAME=>'Earnings',
P_OWNER=>'Oracle9iAS Portal', P_SERVERS=>'repserver,acct_server',
P_FILENAME=>'earnings.rdf', P_DESCRIPTION=>'restricted report',
P_PRIVILEGE=>'SCOTT,JABERS,ACCT', P_AVAILABILITY=>'production'
P_TYPES=>'Cache,printer', P_FORMATS=>'HTMLCSS,PDF)',
P_PRINTERS=>'sales_printer,acct_printer',
P_PFORMTEMPLATE=>'public.finance_template',
P_TRIGGER=>'Is begin IF UPPER(DESTYPE) = 'PRINTER' AND EMPNAME = 'JABERS'
THEN RETURN(TRUE); ELSE RETURN(FALSE); END IF; end;');

EXEC :STATUS := RWWWVREG.REGISTER_REPORT (P_NAME=>'Acc_pay',
P_OWNER=>'Oracle9iAS Portal', P_SERVERS=>'repserver,acct_server',
P_FILENAME=>'acc_pay.rdf', P_DESCRIPTION=>'restricted report',
P_PRIVILEGE=>'SCOTT,JABERS,ACCT', P_AVAILABILITY=>'production'
P_TYPES=>'Cache,printer', P_FORMATS=>'HTMLCSS,PDF',
p_printers=>'sales_printer,acct_printer',
P_PFORMTEMPLATE=>'public.finance_template'
P_TRIGGER=>'Is begin IF UPPER(DESTYPE) = 'PRINTER' AND
EMPNAME = 'JABERS' THEN RETURN(TRUE); ELSE RETURN(FALSE); END IF; end;');

```

この SQL スクリプト・ファイルの内容の詳細は、「[PL/SQL バッチ登録ファンクション](#)」を参照してください。

2. **reports.log** ファイルで、変換処理中に発生したエラーがないかどうかを確認します。このファイルは通常、現在作業中のディレクトリに作成されます。**reports.log** ファイルが生成されなかった場合、**RWCONVERTER** で発生したエラーがないことを示します。
3. ここで、システム・パラメータおよびユーザー・パラメータの値を必要に応じて編集できます。たとえば、前述のサンプル・スクリプトで最初の **RWWWVREG** ファンクションにより、**P_PARAMETERS** という追加パラメータが生成されました。この処理が行われた理由は、**security.rdf** ファイルに、**P_LASTNAME** と **P_SSN** という 2 つのユーザー定義パラメータが次のように記述されているためです。

```
P_PARAMETERS=>'(P_LASTNAME) (P_SSN)',
```

この場合、デフォルト、下限、上限の各値をオプションで定義したり、各ユーザー・パラメータの値リストを定義して、実行時にユーザーが入力できる値を制限できます。同様に、システム・パラメータを制限する場合には（たとえば、**COPIES** を使用してユーザーが作成できるコピーの数を制限する場合）、**P_PARAMETERS** パラメータを使用します。編集後の **P_PARAMETERS** キーワードは、次のようになります。

```
P_PARAMETERS=>'(P_LASTNAME, LOV=LASTNAME_LOV) (P_SSN)
```

```
(COPIES, DEFAULT=1,LOW=1,HIGH=2)'
```

この修正後のコード・セグメントでは、レポートは次のように制限されます。

- P_LASTNAME ユーザー・パラメータは、LASTNAME_LOV 値リストに含まれる値に制限されます。
 - P_SSN にユーザーが指定する値が、P_LASTNAME 値の評価に必要です。
 - COPIES システム・パラメータのデフォルト値は 1、また印刷部数は 1 か 2 である必要があります。
4. output.sql ファイルを保存してクローズします。

C.1.2 SQL*Plus でのスクリプトの実行

レポートを Oracle9iAS Portal に実際に登録するには、RWCONVERTER により生成したスクリプトを実行する必要があります。

1. SQL*Plus を開始し、パッケージ・プロシージャを所有させる Oracle9iAS Portal スキーマにログインします。
2. SQL*Plus コマンド・プロンプトから、RWCONVERTER で作成したスクリプトを実行します。

```
@ output.sql
```

このスクリプトは、スクリプトに示された各レポートについて、指定されたパラメータを使用して Oracle9iAS Portal でパッケージを実行および作成します。

3. RW_ADMINISTRATOR 権限を持つユーザーとして Oracle9iAS Portal にログインします。
4. 「企業ドキュメント」タブをクリックします。
5. 「ビルダー」をクリックします。
6. 「管理」タブをクリックします。
7. 「Oracle Reports Security」ポートレットで、「Oracle Reports Security の設定」をクリックします。
8. 「レポート定義ファイル・アクセス」ポートレットで、SQL スクリプトでバッチ登録したいいずれかのレポートの P_NAME を入力します。
9. 「編集」をクリックします。「コンポーネントの管理」ページが表示されます。
10. ページ下部の「編集」をクリックして、レポートのパラメータを編集します。
11. パラメータを必要に応じて編集します。
12. 「OK」をクリックします。
13. 「閉じる」をクリックします。

14. スクリプトでバッチ登録したレポートごとにステップ 8～13 を繰り返します。

C.2 レポート・パッケージのバッチ削除

Oracle9iAS Portal から多数のレポートをバッチ削除する手順は、次のとおりです。

1. テキスト・エディタで、削除するレポート定義ファイル・パッケージごとに、1 つの RWWWVREG.DEREGISTER_REPORT ファンクション・コールを含む SQL スクリプト・ファイル（例：rmv_rdfs.sql）を作成します。次に例を示します。

```
VAR STATUS NUMBER;
EXEC :STATUS := RWWWVREG.DEREGISTER_REPORT (P_NAME=>'Security');
EXEC :STATUS := RWWWVREG.DEREGISTER_REPORT (P_NAME=>'Earnings');
EXEC :STATUS := RWWWVREG.DEREGISTER_REPORT (P_NAME=>'Acc_pay');
```

注意： P_NAME は、Oracle9iAS Portal から削除するレポート定義ファイル・パッケージの名前です。

2. SQL*Plus を開始し、レポートのパッケージ・プロシージャを所有する Oracle9iAS Portal スキーマにログインします。
3. SQL*Plus コマンド・プロンプトから、最初のステップで作成したスクリプトを実行します。

```
@ rmv_rdfs.sql
```

このスクリプトは、スクリプトに示された各レポートについて、Oracle9iAS Portal からパッケージを削除します。

注意： このプロシージャを実行しても、レポート定義ファイルはファイル・システムから削除されません。レポートが登録解除され、Oracle9iAS Portal から利用できなくなるだけです。レポート定義ファイルを削除するには、ユーザーがファイル・システムから削除する必要があります。

C.3 PL/SQL バッチ登録ファンクション

レポートを Oracle9iAS にバッチ登録する目的で RWCONVERTER によって生成された SQL スクリプトは、主に rwwwvreg.register_report ファンクションへのコールで構成されています。rwwwvreg.register_report の構文を次に示します。

```
Function Rwwwvreg.register_report (
    p_owner varchar2,
    p_name varchar2,
    p_servers varchar2,
```

```
p_filename varchar2,  
p_description varchar2,  
p_privileges varchar2,  
p_availability varchar2,  
p_types varchar2,  
p_formats varchar2,  
p_printers varchar2,  
p_pdformTemplate varchar2,  
p_parameters varchar2,  
p_trigger varchar2)  
return number;  
-- =0 : succeeded;  
-- !=0 : failed;
```

次の表に、rwwwvreg.register_report が取得するパラメータを示します。

表 C-1 rwwwvreg.register_report のパラメータ

パラメータ	説明
P_OWNER	<p>スキーマの所有者。デフォルトは、SQL*PLUS スクリプト開始時にログインする現在の Oracle9iAS Portal スキーマです。</p> <p>次に例を示します。</p> <pre>P_OWNER=>'Oracle9iAS Portal'</pre>
P_NAME	<p>Oracle9iAS Portal でのレポートの識別に使用される名前。</p> <p>P_NAME は、「レポート定義ファイルのアクセス権限を作成」ウィザードの「名前」フィールドに対応しています。</p> <p>次に例を示します。</p> <pre>P_NAME=>'Earnings'</pre>
P_SERVERS	<p>P_FILENAME パラメータで定義されたレポート定義ファイルがアクセス権を持つ Reports Servers の名前。Reports Servers のリストはカンマで区切ります。</p> <p>P_FILENAME は、「レポート定義ファイルのアクセス権限を作成」ウィザードおよび「Edit Report Definition File」ページの「Reports サーバー」フィールドに対応しています。</p> <p>次に例を示します。</p> <pre>P_SERVERS=>'repserver,acct'</pre> <p>注意: P_SERVERS で示す Reports Server は、事前に Oracle9iAS Portal に登録されている必要があります。詳細は、第 5 章「ユーザー・アクセスの制御」を参照してください。</p>

表 C-1 rwwwvreg.register_reportのパラメータ (続き)

パラメータ	説明
P_FILENAME	<p>登録するレポート定義ファイルの名前。</p> <p>P_FILENAME は、「レポート定義ファイルのアクセス権限を作成」ウィザードおよび「Edit Report Definition File」ページの「Oracle Reports ファイル名」に対応しています。</p> <p>次に例を示します。</p> <pre>P_FILENAME=>'earnings.rdf'</pre>
P_DESCRIPTION	<p>レポートの説明。</p> <p>P_DESCRIPTION は、「レポート定義ファイルのアクセス権限を作成」ウィザードおよび「Edit Report Definition File」ページの「説明」フィールドに対応しています。</p> <p>次に例を示します。</p> <pre>P_DESCRIPTION=>'restricted report'</pre>
P_PRIVILEGE	<p>P_FILENAME で定義されたレポート定義ファイルの実行に必要な権限を持つユーザーまたはロール。このリストはカンマで区切ります。</p> <p>P_PRIVILEGE は、「コンポーネントの管理」ページの「アクセス」タブの、レポートに対する「ユーザー / ロール」リストに対応しています。「ユーザー / ロール」リストを表示するには、「Inherit Privileges from Portal DB Provider」のチェックを外す必要があります。</p> <p>次に例を示します。</p> <pre>P_PRIVILEGE=>'SCOTT,JABERS,PORTAL90'</pre>
P_AVAILABILITY	<p>P_FILENAME パラメータで定義されたレポート定義ファイルがいつ処理できるようになるかを決定する使用可能日カレンダーの名前。</p> <p>P_AVAILABILITY は、「レポート定義ファイルのアクセス権限を作成」ウィザードおよび「Edit Report Definition File」ページの「使用可能日カレンダー名」フィールドに対応しています。</p> <p>次に例を示します。</p> <pre>P_AVAILABILITY=>'production'</pre> <p>注意: 使用可能日カレンダーは、事前に Oracle9iAS Portal に登録されている必要があります。使用可能日カレンダーの作成の詳細は、第 5 章「ユーザー・アクセスの制御」を参照してください。</p>

表 C-1 rwwwvreg.register_report のパラメータ (続き)

パラメータ	説明
P_TYPES	<p>P_FILENAME パラメータで定義されたレポート定義ファイルを送信できる宛先タイプ (キャッシュ、プリンタなど)。このリストはカンマで区切ります。</p> <p>P_TYPES は、「レポート定義ファイルのアクセス権限を作成」ウィザードおよび「Edit Report Definition File」ページの「タイプ」複数選択ボックスに対応しています。</p> <p>次に例を示します。</p> <p>P_TYPES=>'CACHE,printer'</p>
P_FORMATS	<p>P_FILENAME パラメータで定義されたレポート定義ファイルを送信できる宛先フォーマット (HTML、PDF など)。このリストはカンマで区切ります。</p> <p>P_FORMATS は、「レポート定義ファイルのアクセス権限を作成」ウィザードおよび「Edit Report Definition File」ページの「書式」複数選択ボックスに対応しています。</p> <p>次に例を示します。</p> <p>P_FORMATS=>'HTMLCSS,PDF'</p>
P_PRINTERS	<p>P_FILENAME パラメータで定義されたレポート定義ファイルを印刷できるプリンタ。このリストはカンマで区切ります。</p> <p>P_PRINTERS は、「レポート定義ファイルのアクセス権限を作成」ウィザードおよび「Edit Report Definition File」ページの「プリンタ」複数選択ボックスに対応しています。</p> <p>次に例を示します。</p> <p>P_PRINTERS=>'sales_printer,acct_printer'</p> <p>注意: P_PRINTERS で示すプリンタは、事前に Oracle9iAS Portal に登録されている必要があります。詳細は、第 5 章「ユーザー・アクセスの制御」を参照してください。</p>
P_PFORMTEMPLATE	<p>ランタイム・パラメータ・フォームのページ・スタイルを決定するパラメータ・フォーム・テンプレート。</p> <p>P_PFORMTEMPLATE は、「レポート定義ファイルのアクセス権限を作成」ウィザードおよび「Edit Report Definition File」ページの「パラメータ・フォーム・テンプレート」フィールドに対応しています。</p> <p>次に例を示します。</p> <p>P_PFORMTEMPLATE=>'public.finance_template'</p>

表 C-1 rwwwvreg.register_reportのパラメータ (続き)

パラメータ	説明
P_PARAMETERS	<p>ユーザー・パラメータおよびシステム・パラメータのデフォルト値、上限値および下限値、あるいは値リストの名前。</p> <p>注意: P_PARAMETERS パラメータには対応する RWCONVERTER 引数がありません。このため、ユーザー・パラメータの値、範囲、または値リストをバッチ・インポートする場合には、RWCONVERTE によって生成された SQL スクリプトを手動で編集する必要があります。</p> <p>P_PARAMETERS は、「レポート定義ファイルのアクセス権限を作成」ウィザードおよび「Edit Report Definition File」ページの「(パラメータの) 名前」、「LOV」、「下限」および「上限」フィールドに対応しています。</p> <p>デフォルトは、指定されたパラメータのランタイム・パラメータ・フォームに設定されている値に対応しています。</p> <p>次に例を示します。</p> <pre>P_PARAMETERS=>' (P_LASTNAME, LOV=LASTNAME_LOV) (P_SSN) (COPIES, DEFAULT=1,LOW=1,HIGH=2) '</pre> <p>各項目の内容は次のとおりです。</p> <p>P_LASTNAME、P_SSN および COPIES は、パラメータ名です。</p> <p>LOV は、値リスト名です。</p> <p>DEFAULT は、デフォルト値です。</p> <p>LOW は、値範囲の下限です。</p> <p>HIGH は、値範囲の上限です。</p>
P_TRIGGER	<p>PL/SQL で記述された有効化トリガーで、ブール文 (TRUE (成功した場合) または FALSE (失敗した場合)) を返します。</p> <p>P_TRIGGER は、「レポート定義ファイルのアクセス権限を作成」ウィザードおよび「Edit Report Definition File」ページのテキスト・ボックスに対応しています。</p> <p>次に例を示します。</p> <pre>P_TRIGGER=>'Is begin IF UPPER(DESTYPE) = ''PRINTER'' AND EMPNAME = ''SMITH'' THEN RETURN(TRUE); ELSE RETURN(FALSE); END IF; end;'</pre>

索引

A

accessible コマンド・キーワード, A-7
ALTER_SESSION, 12-15
API
 cache, 3-7
 clients, 3-22
 destinations, 3-13
 engine, 3-8
 events, 3-14, 11-1 ~ 11-13
 notification, 3-15
 Pluggable Destinations, 4-2, 9-1, 9-34
 repository, 3-18
 security, 3-11
 Web Object Cache, 14-9
 デバッグ・イベント, 11-7
arraysize コマンド・キーワード, A-8
attach 配布要素, 9-12
 format 属性, 9-12
 instance 属性, 9-13
 name 属性, 9-12
 srcType 属性, 9-12
authid コマンド・キーワード, 2-5, A-8
authid パラメータ、イベント, 11-5
autocommit コマンド・キーワード, A-9
autostart コマンド・キーワード, 2-2, A-9

B

batch コマンド・キーワード, 2-2, 2-3, 2-5, A-10
bcc コマンド・キーワード, A-10
bcc 属性、mail, 9-8
blankpages コマンド・キーワード, A-11
body 配布要素, 9-10
 format 属性, 9-11

instance 属性, 9-11
srcType 属性, 9-11
buffers コマンド・キーワード, A-12

C

cache, 3-7
cacheDir, 3-8
cachelob コマンド・キーワード, A-12
cacheSize 属性, 14-7
cacheSize プロパティ, 3-8
cache 宛先タイプ, 4-5
cache 構成要素, 3-7, 8-12, 14-7
 cacheSize 属性, 14-7
 class 属性, 3-7
callBackTimeOut 属性, 3-10, 14-5
cc コマンド・キーワード, A-13
cc 属性、mail, 9-8
cellwrapper コマンド・キーワード, A-13
CGI, 1-2, 1-6
 URL 構文, 8-4
 下位互換性, 3-6
cgicmd.dat, 3-28, 8-12, 8-15
 エントリの追加, 8-14
 使用, 8-14
classPath 属性、engine, 3-9
class 属性
 cache, 3-7
 destination, 3-13
 engine, 3-9
 jobStatusRepository, 3-18
 notification, 3-16
 security, 3-12, 5-3
cluster 構成要素, 3-22
cmdfile コマンド・キーワード, A-14

cmdkey コマンド・キーワード, A-15
cmdkey パラメータ、イベント, 11-9
compatible 構成要素, 3-3, 3-5
 version 属性, 3-6
confidential 属性, 4-6, 5-4
connection 構成要素, 3-21
 idleTimeOut 属性, 3-21
 maxConnect 属性, 3-21
contentarea コマンド・キーワード, A-16
Cookie, 7-2
cookieexpire、rwsvrlet.properties, 3-32
copies コマンド・キーワード, A-17
copies 属性
 printer, 9-20
Custom Tag Handler, 1-6
customize キーワード, 10-16
customize コマンド・キーワード, 10-1, 10-3, 10-21,
 10-23, A-18

D

DAS、「Delegated Administration Service」を参照
dateformatmask コマンド・キーワード, A-18
db_sys_diffauth、rwsvrlet.properties, 3-29
db_sys_sameauth、rwsvrlet.properties, 3-29
dbauth、rwsvrlet.properties, 3-29
dbms_AQadm パッケージ, 11-10
dbms_aq.dequeue, 11-13
delauth コマンド・キーワード, A-19
Delegated Administration Service, 5-6, 7-1, 7-4
delimited_hdr コマンド・キーワード, A-20
delimiter コマンド・キーワード, A-20
DEQUEUE, 11-10
desformat コマンド・キーワード, A-21
desname コマンド・キーワード, A-22
destinations 配布要素, 9-4
destination 構成要素, 3-12, 4-5
 class 属性, 3-13
 destype 属性, 3-13
destination コマンド・キーワード, A-24
destype コマンド・キーワード, A-25
destype 属性, 3-13, 4-5
 cache, 4-5
 file, 4-5
 mail, 4-5
 oraclePortal, 4-5
 printer, 4-5

destype 配布要素, 9-20
 id 属性, 9-21
 instance 属性, 9-22
 name 属性, 9-21
dest コマンド・キーワード, A-23
DEVELOPER_NLS_LANG, 12-9, 12-12, B-2
diagbodytags、rwsvrlet.properties, 3-33
diagheadtags、rwsvrlet.properties, 3-33
diagnostic、rwsvrlet.properties, 3-29
distribute コマンド・キーワード, A-26
distribution.dtd, 9-2, 9-34
distribution.xml, 9-34
dist パラメータ、イベント, 11-9
DTD、「データ型ディクショナリ」を参照
dtype コマンド・キーワード, A-27
dunit コマンド・キーワード, A-28

E

encrypted 属性, 3-12, 4-6, 5-4
encryptionkey、rwsvrlet.properties, 3-33
engineId 属性, 3-15
engine 構成要素, 3-8, 14-3
 callBackTimeOut 属性, 3-10, 14-5
 classPath 属性, 3-9
 class 属性, 3-9
 engLife 属性, 3-9, 14-4
 id 属性, 3-9
 initEngine 属性, 3-9, 14-3
 maxEngine 属性, 3-9, 14-3
 maxIdle 属性, 3-10, 14-4
 minEngine 属性, 3-9, 14-4
engLife 属性, 3-9, 14-4
ENQUEUE, 11-10
errortemplate, 3-31
expiration コマンド・キーワード, 1-8, 14-7, 14-8,
 A-29
expiredays コマンド・キーワード, A-30
express_server コマンド・キーワード, A-30

F

FIFO, 14-7
fileName 属性, 3-24
file 宛先タイプ, 4-5
file 配布要素, 9-16
 format 属性, 9-18

id 属性, 9-18
instance 属性, 9-18
name 属性, 9-18
foreach 配布要素, 9-5
format 属性
 attach, 9-12
 body, 9-11
 file, 9-18
formsize コマンド・キーワード, A-32
from コマンド・キーワード, A-33
from 属性、mail, 9-9

G

gateway パラメータ、イベント, 11-4
getjobid コマンド・キーワード, A-34
getserverinfo コマンド・キーワード, A-34

H

help、rwservlet.properties, 3-33
help コマンド・キーワード, A-35
HTML
 診断の出力, 3-33
 デバッグの出力, 3-33
HTTP Secure Sockets Layer, 1-6
HTTP Server, 1-6, 7-3
HTTPS, 1-6

I

identifier 構成要素, 3-24
idleTimeOut 属性, 3-21
id 属性
 destype, 9-21
 engine, 3-9
 file, 9-18
 mail, 9-8
 notification, 3-16
 orbClient, 3-22
 printer, 9-19
 security, 3-11, 5-3
ignoremargin コマンド・キーワード, A-35
image_url、rwservlet.properties, 3-32
include 配布要素, 9-14
 src 属性, 9-16
initEngine 属性, 3-9, 14-3

install コマンド・キーワード, A-36, A-76
instance 属性
 attach, 9-13
 body, 9-11
 destype, 9-22
 file, 9-18
 printer, 9-20
itemtitle コマンド・キーワード, A-36

J

Java 2 Enterprise Edition, 1-6
Java Servlet, 1-2
jobname コマンド・キーワード, A-37
jobStatusRepository 構成要素, 3-17, 14-17
 class 属性, 3-18
 repositoryconn 属性, 14-17
jobtype コマンド・キーワード, A-37
jobType 属性, 3-14
job 構成要素, 3-14
 engineId 属性, 3-15
 jobType 属性, 3-14
 securityId 属性, 3-15
JSP, 1-2, 1-6
NLS, 12-9, 12-12
taglib, 14-9
URL 構文, 8-3
Web Object Cache, 14-6, 14-9
イメージ, 3-32
キー・マッピングの使用方法, 8-15
キャッシュの設定, 14-9
キャラクタ・セットの指定, 12-9, 12-12

K

KeyMapFile, 3-28
killjobid コマンド・キーワード, A-38

L

LDAP, 5-4, 7-5
log 構成要素, 3-16
 option 属性, 3-17
longchunk コマンド・キーワード, A-38

M

mail 宛先タイプ, 4-5
mail 配布要素, 9-7

- bcc 属性, 9-8
- cc 属性, 9-8
- from 属性, 9-9
- id 属性, 9-8
- organization 属性, 9-9
- priority 属性, 9-9
- replyTo 属性, 9-9
- returnReceipt 属性, 9-9
- subject 属性, 9-9
- to 属性, 9-8

maxConnect 属性, 3-21
maxEngine 属性, 3-9, 14-3
maxIdle 属性, 3-10, 14-4
maxQueueSize 属性, 3-23
MESSAGES, 11-10
minEngine 属性, 3-9, 14-4
mod_oc4j, 1-6
mod_osso, 7-3
mode コマンド・キーワード, A-39
module キーワード, 10-15
module コマンド・キーワード, 10-22, A-40
MyIdent.AuthID, 11-5
MyIdent.GatewayURL, 11-5
MyIdent.JobID, 11-5
MyIdent.ServerName, 11-5

N

name 属性, 3-26

- attach, 9-12
- destype, 9-21
- file, 9-18
- printer, 9-19

NLS, 12-1 ~ 12-16

- ALTER_SESSION, 12-15
- DEVELOPER_NLS_LANG, 12-9
- JSP, 12-9 ~ 12-12
- NLS_LANG, 12-3, 12-13
- Unicode, 12-13, 12-15
- USER_NLS_LANG, 12-9
- アーキテクチャ, 12-2
- アプリケーションの翻訳, 12-15
- 環境変数, 12-2 ~ 12-9

NLS_LANG, 12-13
キャラクタ・セット, 12-6
言語依存データ, 12-2
言語に依存しない機能, 12-2
フォントのサポート, 12-14
NLS_CALENDAR, B-2
NLS_CREDIT, B-2
NLS_CURRENCY, B-2
NLS_DATE_FORMAT, B-2
NLS_DATE_LANGUAGE, B-2
NLS_DEBIT, B-2
NLS_ISO_CURRENCY, B-2
NLS_LANG, 12-3, 12-13, 12-14, B-2

- charset, 12-4
- language, 12-3
- territory, 12-3
- 言語の表記規則, 12-8
- 構文, 12-3
- 地域の表記規則, 12-8
- 定義, 12-6

NLS_LANG 環境変数, 12-13, 12-14
NLS_LIST_SEPARATOR, B-2
NLS_MONETARY_CHARACTERS, B-2
NLS_NUMERIC_CHARACTERS, B-2
NLS_SORT, B-2
nonblocksql コマンド・キーワード, A-40
notification 構成要素, 3-15

- class 属性, 3-16
- id 属性, 3-16

notifyfailure コマンド・キーワード, A-41
notifysuccess コマンド・キーワード, A-41
numberformatmask コマンド・キーワード, A-42

O

OC4J, 2-5, 7-4, 14-9
OEM, 13-1, 13-15

- Reports Server の開始, 13-3
- Reports Server の再起動, 13-3
- Reports Server の再構成, 13-12
- Reports Server の停止, 13-3
- Reports Server 用の構成, 3-37
- Reports Services へのナビゲート, 13-3
- 起動, 13-3
- ジョブ・キューの管理, 13-4
- ジョブ・キューの表示, 13-4
- パフォーマンスの監視, 13-11

- OID、「Oracle Internet Directory」を参照
- ojsp
 - cacheInclude タグ, 14-10
 - cacheXMLObj タグ, 14-9
 - cache タグ, 14-9
 - invalidateCache タグ, 14-10
 - useCacheObj タグ, 14-9
- onfailure コマンド・キーワード, A-42
- onsuccess コマンド・キーワード, A-43
- OPMN, 2-5
- option 属性, 3-17
- Oracle9iAS Portal へのレポートのバッチ登録, C-1
- Oracle Advanced Queuing, 11-10, 11-12
 - dbms_AQadm パッケージ, 11-10
 - dbms_aq.dequeue, 11-13
 - DEQUEUE, 11-10
 - ENQUEUE, 11-10
 - MESSAGES, 11-10
- Oracle Containers for J2EE, 1-6
- Oracle Containers for Java 2 Enterprise Edition, 7-4
- Oracle Delegated Administration Service, 7-6
- Oracle Enterprise Manager、「OEM」を参照
- Oracle HTTP Server, 7-3
- Oracle Internet Directory, 5-4, 7-1, 7-4, 7-5, 7-6
- Oracle Login Server, 7-1, 7-5
- Oracle Program Manager, 2-5
- Oracle Trace, 14-11, 14-12
- ORACLE_HOME, 3-1
- Oracle9iAS HTTP Server, 1-6, 2-1
- Oracle9iAS Portal, 5-1 ~ 5-19
 - RW_ADMINISTRATOR, 5-5
 - RW_BASIC_USER, 5-5
 - RW_DEVELOPER, 5-5
 - RW_POWER_USER, 5-5
 - アクセス制御, 5-6
 - 概要, 5-2
 - サーバー・アクセス, 5-13
 - 使用可能日カレンダー
 - 結合, 5-9
 - シンプル, 5-7
 - プリンタ・アクセス, 5-11
 - ユーザーとグループ, 5-4
 - ランタイム・パラメータ・フォーム, 5-19
 - レポート・アクセス, 5-15
 - レポートのバッチ登録, C-1
 - レポート・ポートレットの公開, 8-7
 - レポート要求, 8-6

- Oracle9iAS Reports Engine、クラスタ化のアーキテクチャ、ロード・バランシング, 6-2
- Oracle9iAS Reports Services
 - Single Sign-On, 5-2
 - セキュリティの構成, 5-4
 - 対象読者, xvi
- Oracle9iAS Single Sign-On, 5-2
- oraclePortal 宛先タイプ, 4-5
- orbClient 構成要素, 3-21
 - id 属性, 3-22
 - publicKeyFile 属性, 3-22
- organization 属性、mail, 9-9
- orientation コマンド・キーワード, A-44
- outputfolder コマンド・キーワード, A-44
- outputpage コマンド・キーワード, A-45
- overwrite コマンド・キーワード, A-46

P

- p_availability コマンド・キーワード, A-47
- p_description コマンド・キーワード, A-47
- p_formats コマンド・キーワード, A-48
- p_name コマンド・キーワード, A-48
- p_owner コマンド・キーワード, A-49
- p_pformtemplate コマンド・キーワード, A-49
- p_printers コマンド・キーワード, A-50
- p_privilege コマンド・キーワード, A-50
- p_servers コマンド・キーワード, A-51
- p_trigger コマンド・キーワード, A-51
- p_types コマンド・キーワード, A-52
- pagegroup コマンド・キーワード, A-52
- pagesize コマンド・キーワード, A-53
- pagestream コマンド・キーワード, A-54
- paramform コマンド・キーワード, A-55
- ParamList-Object, 11-2
- ParamList-Type, 11-2
- parsequery コマンド・キーワード, A-55
- PATH, 2-4
- pdfcomp コマンド・キーワード, A-56
- pdfembed コマンド・キーワード, A-56
- persisiFile 構成要素
 - fileName 属性, 3-24
- persistFile 構成要素, 3-3, 3-24
- PL/SQL
 - ALTER_SESSION, 12-15
 - SRW_PARAMETER, 11-3
 - SRW_PARAMLIST, 11-3, 11-5, 11-10

- srw_test.sql, 11-8
- SRW.ADD_DEFINITION, 10-17, 10-23
- SRW.ADD_PARAMETER, 11-3
- srwAPIdrop.sql, 11-2
- srwAPIgrant.sql, 11-2
- srwAPIins.sql, 11-2
- SRW.APPLY_DEFINITION, 10-3, 10-15, 10-17
- SRW.CANCEL_REPORT, 11-7
- SRW.CLEAR_PARAMETER_LIST, 11-4
- SRW.JOB_IDENT, 11-5
- SRW-Package, 11-2
- SRW.REMOVE_PARAMETER, 11-4
- SRW.REPORT_STATUS, 11-5
- SRW.START_DEBUGGING, 11-7
- SRW.STATUS_RECORD, 11-6
- SRW.STOP_DEBUGGING, 11-7
 - ブロックの翻訳, 12-15
- PL/SQL、拡張配布, 9-4
- Pluggable
 - clients, 3-22
 - Destinations, 9-34
- pluginParam
 - jobStatusRepository とともに使用, 3-18
 - notification とともに使用, 3-16
- pluginParam 構成要素, 3-25, 9-9
 - cache とともに使用, 3-8
 - destination とともに使用, 3-14
 - engine とともに使用, 3-11
 - name 属性, 3-26
 - security とともに使用, 3-12
 - type 属性, 3-26
- Portal, 5-1 ~ 5-19
 - RW_ADMINISTRATOR, 5-5
 - RW_BASIC_USER, 5-5
 - RW_DEVELOPER, 5-5
 - RW_POWER_USER, 5-5
 - アクセス制御, 5-6
 - 概要, 5-2
 - サーバー・アクセス, 5-13
 - 使用可能日カレンダー、結合, 5-9
 - 使用可能日カレンダー、シンプル, 5-7
 - プリンタ・アクセス, 5-11
 - ユーザーとグループ, 5-4
 - ランタイム・パラメータ・フォーム, 5-19
 - レポート・アクセス, 5-15
 - レポート要求, 8-6
- PostScript、出力先, 2-3

- printer 宛先タイプ, 4-5
- printer 配布要素, 9-18
 - copies 属性, 9-20
 - id 属性, 9-19
 - instance 属性, 9-20
 - name 属性, 9-19
- printjob コマンド・キーワード, A-57
- priority 属性、mail, 9-9
- property 配布要素, 9-22
- publicKeyFile 属性, 3-22

Q

- Queue Manager, 1-2, 8-10
- Queue Viewer, 1-2
- queue 構成要素
 - maxQueueSize 属性, 3-23
- queue 要素, 3-23

R

- readonly コマンド・キーワード, A-57
- reload_keymap、rwservlet.properties, 3-28
- replaceitem コマンド・キーワード, A-58
- replyto コマンド・キーワード, A-59
- replyTo 属性、mail, 9-9
- Reports
 - URL 構文, 8-2
- Reports CGI, 1-6
- Reports Engine, 1-6, 1-8
- Reports JSP, 1-6
- Reports Queue Manager, 1-2, 8-10
- Reports Queue Viewer, 1-2
- Reports Server, 1-6, 7-4
 - OEM による開始, 13-3
 - OEM による再起動, 13-3
 - OEM による再構成, 13-12
 - OEM による停止, 13-3
 - OEM への追加, 13-14
 - アクセス制御, 5-13
 - 宛先処理, 4-3
 - 宛先タイプの登録, 4-4
 - インプロセス・サーバー, 14-2
 - 永続性, 3-24, 8-12
 - クラスタ, 6-1, 6-6, 14-5, 14-6
 - 構成ファイル, 1-7, 3-2
 - コマンドラインからの起動, 2-3

- サービスとしての起動, 2-2
- サブレットとしての起動, 2-3
- 再起動, 6-6
- ステータス・レコード, 11-6
- チューニング, 14-1 ~ 14-17
- 名前の変更, 6-3
- パフォーマンスの監視, 13-11, 14-10, 14-13
- Reports Services
 - cache API, 3-7
 - clients API, 3-22
 - destinations API, 3-13, 4-2, 9-1
 - engine API, 3-8
 - events API, 3-14
 - notification API, 3-15
 - Reports Engine, 14-2
 - repository API, 3-18
 - security API, 3-11
 - Single Sign-On, 5-2
 - アーキテクチャ, 1-1
 - 永続性, 1-2
 - 監視, 13-1 ~ 13-15
 - 管理, 13-1, 13-15
 - 概要, 1-2
 - 起動と停止, 2-1
 - コンポーネント, 1-5
- Reports Services の監視, 13-1, 13-15
- Reports Services の管理, 13-1, 13-15
- Reports Servlet, 1-6, 3-3, 7-4
 - URL 構文, 8-2
- Reports キャッシュ, 1-6
- REPORTS_CGIDIAGBODYTAGS, B-3
- REPORTS_CGIDIAGHEADTAGS, B-3
- REPORTS_CGIHELP, B-3
- REPORTS_CGIMAP, B-3
- REPORTS_CGINODIAG, B-4
- REPORTS_COOKIE_EXPIRE, B-2
- REPORTS_DB_AUTH, B-2
- REPORTS_ENCRYPTION_KEY, B-2
- REPORTS_PATH, B-4
- REPORTS_SERVER, B-4
- REPORTS_SSLPORT, B-4
- REPORTS_SYS_AUTH, B-4
- REPORTS_TMP, B-4
- reports.dtd, 10-2
- report キーワード, 10-15
- report コマンド・キーワード, 10-22, A-40
- report パラメータ、イベント, 11-4, 11-9
- repositoryconn 属性, 14-17
- returnReceipt 属性、mail, 9-9
- role コマンド・キーワード, A-59
- rundebug コマンド・キーワード, A-60
- RW_ADMINISTRATOR, 5-5
- RW_BASIC_USER, 5-5
- RW_DEVELOPER, 5-5
- RW_POWER_USER, 5-5
- RW_SERVER_QUEUE 表, 14-14
- rw_server.sql, 14-17
- rwbuilder.conf, 3-2
- rwbuilder コマンド, 10-22, A-3
 - 使用するキーワード, A-3
- rwcgi コマンド, A-6
 - 使用するキーワード, A-6
- rwclient, 8-5
- rwclient コマンド, 10-16, 10-20, A-2
 - 使用するキーワード, A-2
- rwconverter
 - バッチ登録用 SQL スクリプトの生成, C-2
- rwconverter コマンド, 10-15, 10-21, A-4
 - 使用するキーワード, A-5
- rwproxy, 3-6
- rwrn.jar, 6-5
- rwrn コマンド, 10-16, 10-17, 10-20, 10-22, A-3
 - 使用するキーワード, A-3
- rwserver, 2-5
 - install, 2-2
 - server, 2-3
- rwserverconf.dtd, 3-2, 3-4, 4-4, 4-6
 - cache 要素, 3-7
 - cluster 要素, 3-22
 - compatible 構成要素, 3-5
 - connection 要素, 3-21
 - destination 要素, 3-12
 - engine 要素, 3-8
 - identifier 要素, 3-24
 - jobStatusRepository 要素, 3-17
 - job 要素, 3-14
 - log 要素, 3-16
 - notification 要素, 3-15
 - orbClient 要素, 3-21
 - persistFile 要素, 3-24
 - pluginParam 要素, 3-25
 - queue 要素, 3-23
 - security 要素, 3-11
 - server 要素, 3-5

- trace 要素, 3-19
- rwserver.template, 3-3
- rwserver コマンド, A-6
 - 使用するキーワード, A-6
- rwervlet, 1-6, 2-3, 14-2
- rwervlet.properties, 3-3, 14-2
 - cookieexpire, 3-32
 - db_sys_diffauth, 3-29
 - db_sys_sameauth, 3-29
 - dbauth, 3-29
 - diagbodytags, 3-33
 - diagheadtags, 3-33
 - diagnostic, 3-29
 - encryptionkey, 3-33
 - errortemplate, 3-31
 - help, 3-33
 - image_url, 3-32
 - KeyMapFile, 3-28
 - reload_keymap, 3-28
 - server, 3-32
 - server_in_process, 3-31
 - sslport, 3-33
 - sysauth, 3-29
 - tracefile, 3-30
 - tracemode, 3-30
 - traceopts, 3-30
 - シングル・サインオン, 3-34
- rwervlet コマンド, A-5
 - 使用するキーワード, A-5
- RW 環境変数, B-4

S

- save_rdf コマンド・キーワード, A-60
- schedule コマンド・キーワード, 8-10, A-61
- Secure Sockets Layer, 1-6
- securityId 属性, 3-15
- security 構成要素, 3-11, 5-2
 - class 属性, 3-12
 - id 属性, 3-11
- security 要素
 - class 属性, 5-3
 - id 属性, 5-3
- server、rwervlet.properties, 3-32
- server、showjobs のパラメータ, 14-13
- server_in_process, 3-31
- server_name.conf, 3-2

- SERVEROUT, 11-7
- server 構成要素, 3-5
- server コマンド・キーワード, 2-3, 2-5, A-62
- server パラメータ、イベント, 11-4
- showenv コマンド・キーワード, A-63
- showjobs コマンド・キーワード, 1-2, 11-6, 14-13,
 - A-63
 - server パラメータ, 14-13
 - statusformat パラメータ, 14-13
- showmap コマンド・キーワード, A-64
- showmyjobs コマンド・キーワード, A-64
- shutdown コマンド・キーワード, 2-5, A-65
- Single Sign-On, 5-2
 - 機能, 5-2
- sitename コマンド・キーワード, A-65
- SMTP, 3-16, 9-8, 9-9
- source コマンド・キーワード, A-66
- SQL*PLUS, 11-7
- srcType 属性
 - attach, 9-12
 - body, 9-11
- src 属性、include, 9-16
- SRW_PARAMETER, 11-3
- SRW_PARAMLIST, 11-3, 11-5, 11-10
- srw_test.sql, 11-8
- SRW.ADD_DEFINITION, 10-17, 10-23
- SRW.ADD_PARAMETER, 11-3
- srwAPIdrop.sql, 11-2
- srwAPIgrant.sql, 11-2
- srwAPIlins.sql, 11-2
- SRW.APPLY_DEFINITION, 10-3, 10-15, 10-17
- SRW.CANCEL_REPORT, 11-7
- SRW.CLEAR_PARAMETER_LIST, 11-4
- SRW.JOB_IDENT, 11-5
- SRW-Package, 11-2
- SRW.REMOVE_PARAMETER, 11-4
- SRW.REPORT_STATUS, 11-5
- SRW.RUN_REPORT, 8-6
- SRW.START_DEBUGGING, 11-7
- SRW.STATUS_RECORD, 11-6
- SRW.STOP_DEBUGGING, 11-7
- SSL, 1-6, 3-33
- sslport、rwervlet.properties, 3-33
- SSO, 3-34
- SSO、「シングル・サインオン」を参照
- ssoconn コマンド・キーワード, A-67
- statusfolder コマンド・キーワード, A-69

statusformat、showjobs のパラメータ, 14-13
statusformat コマンド・キーワード, A-69
statuspage コマンド・キーワード, A-70
stype コマンド・キーワード, A-71
subject コマンド・キーワード, A-71
subject 属性、mail, 9-9
sysauth、rwservlet.properties, 3-29

T

taglib, 14-9
targets.xml, 13-2, 13-14
tnsnames.ora, 3-6
tolerance コマンド・キーワード, 1-8, 8-11, 8-12, 14-7, 14-8, A-72
to 属性、mail, 9-8
trace_all, 3-20
trace_app, 3-20
trace_brk, 3-20
trace_dbg, 3-20
trace_dst, 3-20
trace_err, 3-20
trace_inf, 3-20
trace_log, 3-20
trace_pls, 3-20
trace_prf, 3-20
trace_sql, 3-20
trace_sta, 3-20
trace_tms, 3-20
trace_wrn, 3-20
tracefile コマンド・キーワード, A-73
tracefile サブレット・パラメータ, 3-30
traceFile 属性, 3-19
tracemode コマンド・キーワード, A-74
tracemode サブレット・パラメータ, 3-30
traceMode 属性, 3-20
traceopts コマンド・キーワード, A-74
traceopts サブレット・パラメータ, 3-30
traceOpts 属性, 3-19
trace 構成要素, 3-19
 traceFile 属性, 3-19
 traceMode 属性, 3-20
 traceOpts 属性, 3-19
TrueType Big Font, 12-14
type 属性, 3-26

U

uifont.ali, 12-7
Unicode, 12-13 ~ 12-15
URL、ランタイム構文, 8-2
urlparameter コマンド・キーワード, A-76
URL エンジン
 構成, 3-34
 要求の送信, 8-10
 要素, 3-34
URL 構文, 8-9
URL ジョブ要求, 8-1, 8-15
USER_NLS_LANG, 12-9, 12-12, B-5
userid コマンド・キーワード, A-77
userid パラメータ、イベント, 11-4, 11-9
UTF8, 12-13

V

version 属性, 3-6

W

Web Object Cache, 14-6, 14-9
webserver_debug コマンド・キーワード, A-77
webserver_docroot コマンド・キーワード, A-78
webserver_port コマンド・キーワード, A-79
Web リスナー, 1-6, 7-3
Windows のレジストリ, 12-6

X

XML

1 つの定義の適用, 10-16
distribution.dtd, 9-2, 9-34
PL/SQL での定義の適用, 10-17
PL/SQL による適用, 10-17
Reports Builder でのオープン, 10-22
reports.dtd, 10-2
targets.xml, 13-2, 13-14
解析, 10-2
カスタマイズの作成, 10-3 ~ 10-8
カスタマイズの適用, 10-3
カスタマイズのデバッグ, 10-21 ~ 10-23
カスタマイズのトレース・オブション, 10-22
クロス積グループの作成, 10-12
グループ階層の作成, 10-11

- サマリーの作成, 10-13
- 式の作成, 10-13
- 書式マスクの変更, 10-5
- 書式例外の追加, 10-5
- 新規問合せの追加, 10-7
- 実行時の適用, 10-15
- スタイルの変更, 10-4
- 単独での実行, 10-20
- 適用, 10-15 ~ 10-21
- データ・ソースのリンク, 10-10
- データ・モデルの作成, 10-9 ~ 10-15
- デバッグ, 10-23
- ハイパーリンクの追加, 10-6
- 配布 XML ファイルの使用, 9-33
- バッチ変更, 10-21
- パーサーのエラー・メッセージ, 10-21
- パラメータの作成, 10-14
- 必須カスタマイズ・タグ, 10-4
- 複数の定義の適用, 10-16
- 複数のデータ・ソースの作成, 10-9
- ブレースホルダの作成, 10-13
- プログラム・ユニットの追加, 10-6
- マトリックス・グループの作成, 10-12
- レポートのカスタマイズ, 10-1 ~ 10-23
- XML、拡張配布, 9-1, 9-33
- XML 属性、変数の使用, 9-3
- XSL、distribution.xsl, 9-34

あ

- アーキテクチャ
 - NLS, 12-2
 - Reports Services, 1-1
 - 宛先, 4-3
 - シングル・サインオン, 7-3
- アクセス制御, 5-6 ~ 5-19
 - サーバー, 5-13
 - 使用可能日カレンダー、結合, 5-9
 - 使用可能日カレンダー、シンプル, 5-7
 - プリンタ, 5-11
 - レポート, 5-15
- 宛先
 - 宛先タイプ, 4-5
 - クラス, 4-5
 - 有効な値, 4-5
- 宛先アーキテクチャ, 4-3
- 宛先タイプ, 3-13, 4-4

- アドバンスト・キューイング, 11-2, 11-10, 11-11, 11-12
 - dbms_AQadm パッケージ, 11-10
 - dbms_aq.dequeue, 11-13
 - DEQUEUE, 11-10
 - ENQUEUE, 11-10
 - MESSAGES, 11-10
- アプリケーションの翻訳, 12-15
- 暗号化, 4-6, 5-4
- 一括処理, 9-6
- イベント
 - authid パラメータ, 11-5
 - cmdkey パラメータ, 11-9
 - dist パラメータ, 11-9
 - gateway パラメータ, 11-4
 - MyIdent.AuthID, 11-5
 - MyIdent.GatewayURL, 11-5
 - MyIdent.JobID, 11-5
 - MyIdent.ServerName, 11-5
 - ParamList-Object, 11-2
 - ParamList-Type, 11-2
 - report パラメータ, 11-4, 11-9
 - server パラメータ, 11-4
 - SRW_PARAMETER, 11-3
 - SRW_PARAMLIST, 11-3, 11-5, 11-10
 - srw_test.sql, 11-8
 - SRW.ADD_PARAMETER, 11-3
 - srwAPIdrop.sql, 11-2
 - srwAPIgrant.sql, 11-2
 - srwAPIlins.sql, 11-2
 - SRW.CANCEL_REPORT, 11-7
 - SRW.CLEAR_PARAMETER_LIST, 11-4
 - SRW.JOB_IDENT, 11-5
 - SRW-Package, 11-2
 - SRW.REMOVE_PARAMETER, 11-4
 - SRW.REPORT_STATUS, 11-5
 - SRW.START_DEBUGGING, 11-7
 - SRW.STATUS_RECORD, 11-6
 - SRW.STOP_DEBUGGING, 11-7
 - userid パラメータ, 11-4, 11-9
 - エンキュー・プロシージャの作成, 11-11
 - ジョブの取消し, 11-6
 - デキュー・プロシージャの作成, 11-12
 - デバッグ, 11-7
 - メッセージ・キューの作成, 11-10
 - レポートの起動, 11-8
- イベント・ドリブンによる公開, 1-9, 11-1 ~ 11-13

インプロセス・サーバー, 1-2, 1-6, 3-31, 14-2
永続性, 1-2, 3-24, 8-12
エラー・メッセージ, 3-29, 3-31
エラー・メッセージ、XML, 10-21
エンキュー、プロシージャの作成, 11-11
エンジン、チューニング, 14-2
大文字と小文字の区別, 2-4

か

各国語サポート、「NLS」を参照

環境変数

DEVELOPER_NLS_LANG, 12-9, B-2
NLS, 12-14
NLS_CALENDAR, B-2
NLS_CREDIT, B-2
NLS_CURRENCY, B-2
NLS_DATE_FORMAT, B-2
NLS_DATE_LANGUAGE, B-2
NLS_DEBIT, B-2
NLS_ISO_CURRENCY, B-2
NLS_LANG, 12-3, B-2
NLS_LIST_SEPARATOR, B-2
NLS_MONETARY_CHARACTERS, B-2
NLS_NUMERIC_CHARACTERS, B-2
NLS_SORT, B-2
PATH, 2-4
REPORTS_CGIDIAGBODYTAGS, B-3
REPORTS_CGIDIAGHEADTAGS, B-3
REPORTS_CGIHELP, B-3
REPORTS_CGIMAP, B-3
REPORTS_CGINODIAG, B-4
REPORTS_COOKIE_EXPIRE, B-2
REPORTS_DB_AUTH, B-2
REPORTS_ENCRYPTION_KEY, B-2
REPORTS_PATH, B-4
REPORTS_SERVER, B-4
REPORTS_SSLPORT, B-4
REPORTS_SYS_AUTH, B-4
REPORTS_TMP, B-4
RW, B-4
USER_NLS_LANG, 12-9, B-5

環境変数、NLS, 12-2, 12-9

環境変数、編集, B-1

関連文書, xvii

キー・マップ・ファイル, 8-12 ~ 8-15

URL パラメータのマッピング, 8-14

位置の指定, 3-28

エントリの追加, 8-14

再ロード, 3-28

使用, 8-14

使用可能にする方法, 8-13

パラメータ・フォームを使用した制限付きの実行
, 8-14

利点, 8-13

キーワード

accessible, A-7
arraysize, A-8
authid, A-8
autocommit, A-9
autostart, A-9
batch, A-10
bcc, A-10
blankpages, A-11
buffers, A-12
cachelob, A-12
cc, A-13
cellwrapper, A-13
cmdfile, A-14
cmdkey, A-15
contentarea, A-16
copies, A-17
customize, 10-1, 10-3, 10-15, 10-16, 10-21, 10-23,
A-18
dateformatmask, A-18
delauth, A-19
delimited_hdr, A-20
delimiter, A-20
desformat, A-21
desname, A-22
dest, A-23
destination, A-24
destype, A-25
distribute, A-26
dtype, A-27
dunit, A-28
expiration, 14-7, 14-8, A-29
expiredays, A-30
express_server, A-30
formsize, A-32
from, A-33
getjobid, A-34
getserverinfo, A-34
help, A-35

ignoremargin, A-35
install, A-36, A-76
itemtitle, A-36
jobname, A-37
jobtype, A-37
killjobid, A-38
longchunk, A-38
mode, A-39
module, 10-15, 10-22, A-40
nonblocksql, A-40
notifyfailure, A-41
notifysuccess, A-41
numberformatmask, A-42
onfailure, A-42
onsuccess, A-43
orientation, A-44
outputfolder, A-44
outputpage, A-45
overwrite, A-46
p_availability, A-47
p_description, A-47
p_formats, A-48
p_name, A-48
p_owner, A-49
p_pformtemplate, A-49
p_printers, A-50
p_privilege, A-50
p_servers, A-51
p_trigger, A-51
p_types, A-52
pagegroup, A-52
pagesize, A-53
pagestream, A-54
paramform, A-55
parsequery, A-55
pdfcomp, A-56
pdfembed, A-56
printjob, A-57
readonly, A-57
replaceitem, A-58
replyto, A-59
report, 10-15, 10-22, A-40
role, A-59
rundebug, A-60
rwbuilder とともに使用, A-3
rwcgi とともに使用, A-6
rwclient とともに使用, A-2

rwconverter とともに使用, A-5
rwruntime とともに使用, A-3
rwserver とともに使用, A-6
rwservlet とともに使用, A-5
save_rdf, A-60
schedule, 8-10, A-61
server, A-62
showenv, A-63
showjobs, 11-6, 14-13, A-63
showmap, A-64
showmyjobs, A-64
shutdown, A-65
sitename, A-65
source, A-66
ssoconn, A-67
statusfolder, A-69
statusformat, A-69
statuspage, A-70
stype, A-71
subject, A-71
tolerance, 8-11, 8-12, 14-7, 14-8, A-72
tracefile, A-73
tracemode, A-74
traceopts, A-74
urlparameter, A-76
userid, A-77
webserver_debug, A-77
webserver_docroot, A-78
webserver_port, A-79
北アフリカ圏の言語, 12-12
キャッシュ, 8-11
FIFO, 14-7
JSP での設定, 14-9
ojsp
 cacheInclude タグ, 14-10
 cacheXMLObj タグ, 14-9
 cache タグ, 14-9
 invalidateCache タグ, 14-10
 useCacheObj タグ, 14-9
Web Object Cache, 14-6, 14-9
 方法, 14-6 ~ 14-10
キャッシュ・キー, 8-11
キャラクタ・セット, 12-1, 12-6
Unicode, 12-13 ~ 12-15
UTF8, 12-13
設計上の考慮事項, 12-6
フォントのエイリアシング, 12-7

キュー

- OEM による管理, 13-4
- OEM による表示, 13-4
- RW_SERVER_QUEUE 表, 14-14
- 現行ジョブ, 13-4
- 失敗したジョブ, 13-9
- 終了したジョブ, 13-7
- スケジュールされたジョブ, 13-5

クラスタ, 3-32, 6-1 ~ 6-6, 14-5 ~ 14-6

- OEM による表示, 13-13
- OEM によるリンク, 13-13
- 概要, 6-2
- セットアップ, 6-3
- 重複ジョブの検出, 8-12
- 要求送信, 6-6

クラスタ化によるフェイルセーフ環境, 14-5

現行ジョブ・キュー, 13-4

言語

- 北アフリカ圏, 12-12
- 中東圏, 12-12

言語、「NLS_LANG」を参照

公開鍵, 6-5

公開鍵ファイル、生成, 3-22

構成, 3-1 ~ 3-34

- OEM, 3-37
- OEM を使用, 13-12
- Reports Services のセキュリティ, 5-4
- rwserver.properties, 14-2
- URL エンジン, 3-34
- 考慮事項, 1-9
- プロキシ情報, 3-36

構成、Reports Server, 1-8

構成ファイル

- Reports Server, 3-2
- rwserverconf.dtd, 3-4
- rwserver.template, 3-3
- rwserverlet.properties, 3-3

構成要素

- cache, 3-7, 8-12, 14-7
- cluster, 3-22
- compatible, 3-3, 3-5
- connection, 3-21
- destination, 3-12, 4-5
- engine, 3-8
- identifier, 3-24
- job, 3-14
- jobStatusRepository, 3-17, 14-17

log, 3-16

notification, 3-15

orbClient, 3-21

persistFile, 3-3, 3-24

pluginParam, 3-25, 9-9

queue, 3-23

security, 3-11, 5-2

server, 3-5

trace, 3-19

構文、Reports の URL, 8-2

構文、コマンド, A-7

コマンド

rwbuilder, 10-22, A-3

rwcgi, A-6

rwclient, 10-16, 10-20, A-2

rwconverter, 10-15, 10-21, A-4

rwrund, 10-16, 10-17, 10-20, 10-22, A-3

rwserver, A-6

rwserverlet, A-5

概要, A-2

構文, A-7

コマンド・キーワード

accessible, A-7

arraysize, A-8

authid, 2-5, A-8

autocommit, A-9

autostart, 2-2, A-9

batch, 2-2, 2-3, 2-5, A-10

bcc, A-10

blankpages, A-11

buffers, A-12

cachelob, A-12

cc, A-13

cellwrapper, A-13

cmdfile, A-14

cmdkey, A-15

contentarea, A-16

copies, A-17

customize, 10-1, 10-3, 10-15, 10-16, 10-21, 10-23, A-18

dateformatmask, A-18

delauth, A-19

delimited_hdr, A-20

delimiter, A-20

desformat, A-21

desname, A-22

dest, A-23

destination, A-24
destype, A-25
distribute, A-26
dtype, A-27
dunit, A-28
expiration, 1-8, 14-7, 14-8, A-29
expiredays, A-30
express_server, A-30
formsize, A-32
from, A-33
getjobid, A-34
getserverinfo, A-34
help, A-35
ignoremargin, A-35
install, A-36, A-76
itemtitle, A-36
jobname, A-37
jobtype, A-37
killjobid, A-38
longchunk, A-38
mode, A-39
module, 10-15, 10-22, A-40
nonblocksql, A-40
notifyfailure, A-41
notifysuccess, A-41
numberformatmask, A-42
onfailure, A-42
onsuccess, A-43
orientation, A-44
outputfolder, A-44
outputpage, A-45
overwrite, A-46
p_availability, A-47
p_description, A-47
p_formats, A-48
p_name, A-48
p_owner, A-49
p_pformtemplate, A-49
p_printers, A-50
p_privilege, A-50
p_servers, A-51
p_trigger, A-51
p_types, A-52
pagegroup, A-52
pagesize, A-53
pagestream, A-54
paramform, A-55

parsequery, A-55
pdfcomp, A-56
pdfembed, A-56
printjob, A-57
readonly, A-57
replaceitem, A-58
replyto, A-59
report, 10-15, 10-22, A-40
role, A-59
rundebug, A-60
rwbuilder とともに使用, A-3
rwcgi とともに使用, A-6
rwclient とともに使用, A-2
rwconverter とともに使用, A-5
rwrund とともに使用, A-3
rwserver とともに使用, A-6
rwservlet とともに使用, A-5
save_rdf, A-60
schedule, 8-10, A-61
server, 2-3, 2-5, A-62
showenv, A-63
showjobs, 1-2, 11-6, 14-13, A-63
showmap, A-64
showmyjobs, A-64
shutdown, 2-5, A-65
sitename, A-65
source, A-66
ssoconn, A-67
statusfolder, A-69
statusformat, A-69
statuspage, A-70
stype, A-71
subject, A-71
tolerance, 1-8, 8-11, 8-12, 14-7, 14-8, A-72
tracefile, A-73
tracemode, A-74
traceopts, A-74
urlparameter, A-76
userid, A-77
webserver_debug, A-77
webserver_docroot, A-78
webserver_port, A-79
コマンドライン、指定, 8-5

さ

サーバー

- rwserver コマンド, A-6
 - インプロセス, 1-2, 1-6, 3-31
- サーバー、インプロセス, 14-2
- サーバー・アクセス制御, 5-13
- サーバー・クラスタ, 3-22, 3-32
- サーバーのクラスタ化, 3-22
- サブレット, 1-2, 1-6, 2-3, 3-3, 7-4
 - rwervlet, 14-2, A-5
 - rwervlet.properties, 14-2
 - URL 構文, 8-2
 - カスタム・ヘルプの追加, 3-33
- 失敗したジョブ・キュー, 13-9
- 終了したジョブ・キュー, 13-7
- 出力
 - PostScript, 2-3
 - プリンタ, 2-3
- 出力処理, 4-1 ~ 4-4
- 出力タイプ, 4-2
- 使用可能日カレンダー, 5-7 ~ 5-11
 - 結合, 5-9
 - シンプル, 5-7
- シングル・サインオン, 7-1 ~ 7-7
 - rwervlet.properties, 3-34
 - 処理, 7-4
- シングルバイト, 12-13
- ジョブ、実行, 8-1 ~ 8-15
- ジョブ・キュー
 - OEM による管理, 13-4
 - OEM による表示, 13-4
 - 失敗, 13-9
 - 終了済み, 13-7
 - スケジュール済み, 13-5
- ジョブの取消し, 11-6
- スクリプト
 - rw_server.sql, 14-17
 - srw_test.sql, 11-8
 - srwAPIIdrop.sql, 11-2
 - srwAPIgrant.sql, 11-2
 - srwAPIins.sql, 11-2
- スケジュールされたジョブ・キュー, 13-5
- ステータス・レコード, 11-6
- セキュリティの構成, 5-4
- セッション Cookie, 7-2
- 接続文字列, 7-5
- 双方向サポート, 12-1, 12-16
- ソート基準、言語, 12-8
- 属性、変数の使用, 9-3

た

- 多言語テキストの表示, 12-14
- 中東圏の言語, 12-12
- 重複ジョブの検出, 8-12
- 月名、言語, 12-8
- テキストの表示、多言語, 12-14
- テキストの読み込み順序, 12-1
- テンプレート、rwserver.template, 3-3
- データ型
 - DATE, 12-3
 - NUMBER, 12-3
- データ型ディクショナリ
 - distribution.dtd, 9-2, 9-34
 - reports.dtd, 10-2
 - rwserverconf.dtd, 3-2, 3-4
- データ・ソース
 - XML によるグループ階層, 10-11
 - XML による作成, 10-9
 - XML によるリンク, 10-10
- データベース・トリガー, 11-8
- データベース認証 Cookie, 3-32, 3-33
- データ・モデル、作成, 10-9, 10-15
- デキュー、プロシージャの作成, 11-12
- デバッグ・イベント, 11-7
- トランスポートブル
 - cache, 3-7
 - destinations, 3-13
 - engine, 3-8
 - events, 3-14
 - notification, 3-15
 - repository, 3-18
 - security, 3-11
 - 宛先, 4-2, 9-1
- トリガー、データベース, 11-8
- トレース, 10-16
- 同等の記号, 12-8
- ドキュメント
 - このマニュアルの構成, xvi

な

- 名前 / 値のペア、宛先, 4-6
- 認証 Cookie, 3-32, 3-33

は

- ハイパーリンク、XML による追加 , 10-6
- ハイパーリンク・ジョブ要求 , 8-9
- 配布 , 9-1 ~ 9-33
 - 一括処理 , 9-6
- 配布、XML ファイルの使用 , 9-33
- 配布の概要 , 9-2
- 配布の例 , 9-23, 9-33
 - file , 9-29
 - foreach , 9-23
 - mail , 9-26
 - printer , 9-30
- 配布要素
 - attach , 9-12
 - body , 9-10
 - destinations , 9-4
 - destype , 9-20
 - file , 9-16
 - foreach , 9-5
 - include , 9-14
 - mail , 9-7
 - printer , 9-18
 - property , 9-22
- バッチ変更、XML , 10-21
- パフォーマンス監視のトレース , 14-11, 14-12
- パフォーマンスの監視 , 13-11, 14-10 ~ 14-13
 - Oracle Trace , 14-11 ~ 14-12
- パラメータ・フォーム , 5-19
 - キー・マップ・ファイル , 8-14
- パラメータ・リスト (イベント)
 - 作成 , 11-3 ~ 11-4
 - 操作 , 11-3, 11-4
- 秘密鍵 , 6-5
- 秘密鍵ファイル、生成 , 3-22
- 表記規則 , xvii
- 表記の方向、言語 , 12-8
- ファイアウォール
 - プロキシ情報 , 3-36
- フォント、TrueType Big , 12-14
- フォントのエイリアシング , 12-7
- フォントのサポート、NLS , 12-14
- フォント・マッピング , 12-7
- プリンタ、出力先 , 2-3
- プリンタ・アクセス制御 , 5-11
- プロキシ情報
 - 構成 , 3-36

- プログラム・ユニット、XML による追加 , 10-6
- プロバイダ
 - レポートに対して作成 , 8-7
- 変数、NLS の環境 , 12-2, 12-9
- 変数、XML 属性での使用 , 9-3
- 方向、言語 , 12-8
- 翻訳
 - PL/SQL ブロック , 12-16
 - TranslationHub ツール , 12-15
- ポートレット
 - ページへの追加 , 8-8
 - レポート定義ファイル・アクセスの作成 , 8-7
 - レポートに対するプロバイダの作成 , 8-7
 - レポートの公開 , 8-7
 - レポートのバッチ登録 , C-1

ま

- マップ
 - URL パラメータ、キー・マップ・ファイル , 8-14
- マルチバイト , 12-1, 12-7, 12-13
- メッセージ、言語 , 12-8
- メッセージ 401 , 7-5
- メッセージ・キュー、作成 , 11-10

や

- 要素、「配布」、「カスタマイズ」または「構成」を参照
- 曜日名、言語 , 12-8
- 読み込み順序 , 12-1, 12-12

ら

- ランタイム URL , 8-1, 8-15
- ランタイム URL 構文 , 8-2
- ランタイム・パラメータ・フォーム , 5-19
- リスナー , 1-6
- 例、配布 , 9-23 ~ 9-33
 - file , 9-29
 - foreach , 9-23
 - mail , 9-26
 - printer , 9-30
- レジストリ、Windows , 12-6
- レジストリ、編集 , B-1
- レポート
 - Oracle9iAS Portal からのバッチ削除 , C-5
 - Oracle9iAS Portal へのバッチ登録 , C-1

- Portal 経由の要求, 8-6
- URL 構文, 8-9
- URL 要求, 8-6
- XML カスタマイズ, 10-3, 10-8
- XML データ・モデル, 10-9, 10-15
- 一括処理, 9-6
- イベントによる起動, 11-8
- カスタム XML の適用, 10-15, 10-21
- カスタム XML のデバッグ, 10-21, 10-23
- キャッシュ, 8-11
- 現行ジョブ・キュー, 13-4
- コマンドライン要求, 8-5
- 失敗したジョブ・キュー, 13-9
- 終了したジョブ・キュー, 13-7
- 処理, 1-7
- 自動実行, 8-10
- スケジューリング, 8-10
- スケジュールされたジョブ・キュー, 13-5
- ハイパーリンクの追加, 8-9
- パッケージ・プロシージャを使用した要求, 8-6
- 要求の方法, 8-5
- レポート、実行, 8-1, 8-15
- レポート・アクセス制御, 5-15
- レポート定義、XML, 10-1
- レポートの実行, 8-1
- レポートの自動実行, 8-10
 - Oracle9iAS Portal から, 8-9
- レポート要求の指定
 - URL エンジン, 8-10
 - Web ブラウザから, 8-9
 - 自動実行のスケジューリング, 8-10
 - レポート・ポートレットの公開, 8-7

