

# Oracle8i

パッケージ・プロシージャ リファレンス Vol. 2

リリース 8.1

部品番号 A62777-1

第 1 版 1999 年 5 月 (第 1 刷)

原本名: Oracle8i Supplied Packages Reference, Volume 2, Release 8.1.5

原本部品番号: A67841-01

原著者: Michele Cyran

原本協力者: D. Alpern, G. Arora, N. Bhatt, S. Chandrasekar, T. Chang, G. Claborn, R. Decker, A. Downing, J. Draaijer, J. Finnerty, R. Frank, A. Ganesh, J. Gosselin, R. Govindarajan, B. Goyal, S. Harris, B. Himatsingka, C. Iyer, H. Jakobsson, A. Jasuja, M. Jungerman, P. Justus, A. Kalra, M. Kamath, S. Kotsovolos, V. Krishnamurthy, J. Krishnaswamy, J. Kundu, B. Lee, J. Liu, P. Locke, A. Logan, N. Mallavarupu, J. Mallory, R. Mani, S. Mavris, A. Mozes, J. Muller, C. Murray, K. Muthukkaruppan, S. Muthulingam, R. Park, G. Pongracz, T. Portfolio, L. Price, S. Puranik, M. Ramacher, S. Ramakrishnan, D. Raphaely, S. Ray, A. Rhee, S. Samu, U. Sangam, A. Saxena, J. Sharma, E. Soylemez, A. Srinivasan, J. Srinivasan, I. Stocks, R. Sujithan, A. Swaminathan, K. Tarkhanov, A. Tsukerman A. To, S. Urman, S. Vivian, D. Voss, W. Wang, R. Ward, S. Wertheimer, R. Wessman, J. Wijaya, D. Wong, L. Wu, R. Yaseen

Copyright © 1999, Oracle Corporation. All rights reserved.

Printed in Japan.

#### 制限付権利の説明

プログラムの使用、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。

当ソフトウェア (プログラム) のリバース・エンジニアリング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 Legend が適用されます。

#### Restricted Rights Legend

Programs delivered subject to the DOD FAR Supplement are "commercial computer software" and use, duplication and disclosure of the Programs 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 52.227-14, Rights in Data -- General, including Alternate III (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 .....	xxxix
<b>1    オラクル社が提供するパッケージ</b>	
パッケージの概要 .....	1-2
オラクル社が提供するパッケージの使用方法 .....	1-2
新規パッケージの作成 .....	1-3
パッケージ内容の参照 .....	1-5
オラクル社が提供するパッケージの一覧 .....	1-6
サブリメンタル・パッケージ内のサブプログラム .....	1-13
カレンダー .....	1-13
SDO_ADMIN .....	1-15
SDO_GEOM .....	1-15
SDO_MIGRATE .....	1-15
SDO_TUNE .....	1-15
TimeScale .....	1-16
TimeSeries .....	1-17
TSTools .....	1-20
UTL_PG .....	1-21
Vir_Pkg .....	1-22
<b>2    DBMS_ALERT</b>	
セキュリティ .....	2-2
定数 .....	2-2
エラー .....	2-2

アラートの使用方法.....	2-3
<b>サブプログラムの要約</b> .....	2-4
REGISTER プロシージャ.....	2-4
REMOVE プロシージャ .....	2-5
REMOVEALL プロシージャ .....	2-5
SET_DEFAULTS プロシージャ.....	2-6
SIGNAL プロシージャ .....	2-6
WAITANY プロシージャ.....	2-7
WAITONE プロシージャ.....	2-8
例.....	2-9

### 3 DBMS\_APPLICATION\_INFO

<b>サブプログラムの要約</b> .....	3-2
SET_MODULE プロシージャ.....	3-2
SET_ACTION プロシージャ.....	3-3
READ_MODULE プロシージャ .....	3-4
SET_CLIENT_INFO プロシージャ.....	3-5
READ_CLIENT_INFO プロシージャ.....	3-6
SET_SESSION_LONGOPS プロシージャ.....	3-6

### 4 DBMS\_AQ

Java クラス .....	4-2
列挙定数.....	4-2
データ構造.....	4-2
<b>サブプログラムの要約</b> .....	4-11
ENQUEUE プロシージャ.....	4-11
DEQUEUE プロシージャ .....	4-13
LISTEN プロシージャ.....	4-15

### 5 DBMS\_AQADM

列挙定数.....	5-2
<b>サブプログラムの要約</b> .....	5-2
CREATE_QUEUE_TABLE プロシージャ.....	5-4
ALTER_QUEUE_TABLE プロシージャ .....	5-7

DROP_QUEUE_TABLE プロシージャ .....	5-8
CREATE_QUEUE プロシージャ .....	5-9
CREATE_NP_QUEUE プロシージャ .....	5-11
ALTER_QUEUE プロシージャ .....	5-12
DROP_QUEUE プロシージャ .....	5-13
START_QUEUE プロシージャ .....	5-14
STOP_QUEUE プロシージャ .....	5-15
GRANT_SYSTEM_PRIVILEGE プロシージャ .....	5-15
REVOKE_SYSTEM_PRIVILEGE プロシージャ .....	5-16
GRANT_QUEUE_PRIVILEGE プロシージャ .....	5-17
REVOKE_QUEUE_PRIVILEGE プロシージャ .....	5-18
ADD_SUBSCRIBER プロシージャ .....	5-18
ALTER_SUBSCRIBER プロシージャ .....	5-19
REMOVE_SUBSCRIBER プロシージャ .....	5-20
SCHEDULE_PROPAGATION プロシージャ .....	5-21
UNSCHEDULE_PROPAGATION プロシージャ .....	5-22
VERIFY_QUEUE_TYPES プロシージャ .....	5-23
ALTER_PROPAGATION_SCHEDULE プロシージャ .....	5-24
ENABLE_PROPAGATION_SCHEDULE プロシージャ .....	5-25
DISABLE_PROPAGATION_SCHEDULE プロシージャ .....	5-26
MIGRATE_QUEUE_TABLE プロシージャ .....	5-27

## 6 DBMS\_DDL

要件 .....	6-2
<b>サブプログラムの要約</b> .....	6-2
ALTER_COMPILE プロシージャ .....	6-2
ANALYZE_OBJECT プロシージャ .....	6-3

## 7 DBMS\_DEBUG

DBMS_DEBUG の使用方法 .....	7-2
使用上の注意 .....	7-5
型 .....	7-6
定数 .....	7-8
エラー・コード .....	7-10

例外 .....	7-11
変数 .....	7-11
<b>サブプログラムの要約 .....</b>	<b>7-12</b>
共通セクション .....	7-13
PROBE_VERSION プロシージャ .....	7-13
SELF_CHECK プロシージャ .....	7-14
SET_TIMEOUT ファンクション .....	7-15
TARGET_SESSION セクション .....	7-15
INITIALIZE ファンクション .....	7-16
DEBUG_ON プロシージャ .....	7-16
DEBUG_OFF プロシージャ .....	7-17
デバッグ・セッション・セクション .....	7-17
ATTACH_SESSION プロシージャ .....	7-18
SYNCHRONIZE ファンクション .....	7-19
SHOW_SOURCE プロシージャ .....	7-19
PRINT_BACKTRACE プロシージャ .....	7-21
CONTINUE ファンクション .....	7-22
SET_BREAKPOINT ファンクション .....	7-23
DELETE_BREAKPOINT ファンクション .....	7-25
DISABLE_BREAKPOINT ファンクション .....	7-25
ENABLE_BREAKPOINT ファンクション .....	7-26
SHOW_BREAKPOINTS プロシージャ .....	7-27
GET_VALUE ファンクション .....	7-28
SET_VALUE ファンクション .....	7-30
DETACH_SESSION プロシージャ .....	7-32
GET_RUNTIME_INFO ファンクション .....	7-32
GET_INDEXES ファンクション .....	7-33
EXECUTE プロシージャ .....	7-34

## 8 DBMS\_DEFER

<b>サブプログラムの要約 .....</b>	<b>8-2</b>
CALL プロシージャ .....	8-2
COMMIT_WORK プロシージャ .....	8-4
datatype_ARG プロシージャ .....	8-4
TRANSACTION プロシージャ .....	8-5

## 9 DBMS\_DEFER\_QUERY

サブプログラムの要約.....	9-2
GET_ARG_FORM ファンクション .....	9-2
GET_ARG_TYPE ファンクション .....	9-3
GET_CALL_ARGS プロシージャ .....	9-5
GET_datatype_ARG ファンクション .....	9-6

## 10 DBMS\_DEFER\_SYS

サブプログラムの要約.....	10-2
ADD_DEFAULT_DEST プロシージャ .....	10-3
DELETE_DEFAULT_DEST プロシージャ .....	10-4
DELETE_DEF_DESTINATION プロシージャ .....	10-4
DELETE_ERROR プロシージャ .....	10-5
DELETE_TRAN プロシージャ .....	10-5
DISABLED ファンクション .....	10-6
EXCLUDE_PUSH ファンクション .....	10-7
EXECUTE_ERROR プロシージャ .....	10-8
EXECUTE_ERROR_AS_USER プロシージャ .....	10-9
PURGE ファンクション .....	10-9
PUSH ファンクション .....	10-11
REGISTER_PROPAGATOR プロシージャ .....	10-13
SCHEDULE_PURGE プロシージャ .....	10-14
SCHEDULE_PUSH プロシージャ .....	10-15
SET_DISABLED プロシージャ .....	10-17
UNREGISTER_PROPAGATOR プロシージャ .....	10-18
UNSCHEDULE_PURGE プロシージャ .....	10-19
UNSCHEDULE_PUSH プロシージャ .....	10-19

## 11 DBMS\_DESCRIBE

セキュリティ .....	11-2
型 .....	11-2
エラー .....	11-2
サブプログラムの要約.....	11-2
DESCRIBE_PROCEDURE プロシージャ .....	11-3

例.....	11-5
--------	------

## 12 DBMS\_DISTRIBUTED\_TRUST\_ADMIN

要件.....	12-2
<b>サブプログラムの要約</b> .....	12-2
ALLOW_ALL プロシージャ.....	12-2
ALLOW_SERVER プロシージャ.....	12-3
DENY_ALL プロシージャ.....	12-4
DENY_SERVER プロシージャ.....	12-4
例.....	12-5

## 13 DBMS\_HS

例外.....	13-2
<b>サブプログラムの要約</b> .....	13-6
ALTER_BASE_CAPS プロシージャ.....	13-8
ALTER_BASE_DD プロシージャ.....	13-9
ALTER_CLASS_CAPS プロシージャ.....	13-9
ALTER_CLASS_DD プロシージャ.....	13-9
ALTER_CLASS_INIT プロシージャ.....	13-9
ALTER_FDS_CLASS プロシージャ.....	13-10
ALTER_FDS_INST プロシージャ.....	13-10
ALTER_INST_CAPS プロシージャ.....	13-10
ALTER_INST_DD プロシージャ.....	13-11
ALTER_INST_INIT プロシージャ.....	13-11
COPY_CLASS プロシージャ.....	13-11
COPY_INST プロシージャ.....	13-12
CREATE_BASE_CAPS プロシージャ.....	13-12
CREATE_BASE_DD プロシージャ.....	13-12
CREATE_CLASS_CAPS プロシージャ.....	13-12
CREATE_CLASS_DD プロシージャ.....	13-13
CREATE_CLASS_INIT プロシージャ.....	13-13
CREATE_FDS_CLASS プロシージャ.....	13-13
CREATE_FDS_INST プロシージャ.....	13-14
CREATE_INST_CAPS プロシージャ.....	13-14



CREATE_INST_DD プロシージャ .....	13-14
CREATE_INST_INIT プロシージャ .....	13-15
DROP_BASE_CAPS プロシージャ .....	13-15
DROP_BASE_DD プロシージャ .....	13-15
DROP_CLASS_CAPS プロシージャ .....	13-15
DROP_CLASS_DD プロシージャ .....	13-16
DROP_CLASS_INIT プロシージャ .....	13-16
DROP_FDS_CLASS プロシージャ .....	13-16
DROP_FDS_INST プロシージャ .....	13-16
DROP_INST_CAPS プロシージャ .....	13-17
DROP_INST_DD プロシージャ .....	13-17
DROP_INST_INIT プロシージャ .....	13-17
REPLACE_BASE_CAPS プロシージャ .....	13-17
REPLACE_BASE_DD プロシージャ .....	13-18
REPLACE_CLASS_CAPS プロシージャ .....	13-18
REPLACE_CLASS_DD プロシージャ .....	13-18
REPLACE_CLASS_INIT プロシージャ .....	13-19
REPLACE_FDS_CLASS プロシージャ .....	13-19
REPLACE_FDS_INST プロシージャ .....	13-19
REPLACE_INST_CAPS プロシージャ .....	13-20
REPLACE_INST_DD プロシージャ .....	13-20
REPLACE_INST_INIT プロシージャ .....	13-21

## 14 DBMS\_HS\_PASSTHROUGH

セキュリティ .....	14-2
<b>サブプログラムの要約</b> .....	14-2
BIND_VARIABLE プロシージャ .....	14-3
BIND_VARIABLE_RAW プロシージャ .....	14-4
BIND_OUT_VARIABLE プロシージャ .....	14-5
BIND_OUT_VARIABLE_RAW プロシージャ .....	14-7
BIND_INOUT_VARIABLE プロシージャ .....	14-8
BIND_INOUT_VARIABLE_RAW プロシージャ .....	14-9
CLOSE_CURSOR プロシージャ .....	14-10
EXECUTE_IMMEDIATE プロシージャ .....	14-11
EXECUTE_NON_QUERY ファンクション .....	14-12

FETCH_ROW ファンクション .....	14-13
GET_VALUE プロシージャ .....	14-14
GET_VALUE_RAW プロシージャ .....	14-15
OPEN_CURSOR ファンクション .....	14-16
PARSE プロシージャ .....	14-17

## 15 DBMS\_IOT

サブプログラムの要約 .....	15-2
BUILD_CHAIN_ROWS_TABLE プロシージャ .....	15-2
BUILD_EXCEPTIONS_TABLE プロシージャ .....	15-3

## 16 DBMS\_JOB

要件 .....	16-2
サブプログラムの要約 .....	16-2
SUBMIT プロシージャ .....	16-3
REMOVE プロシージャ .....	16-4
CHANGE プロシージャ .....	16-5
WHAT プロシージャ .....	16-6
NEXT_DATE プロシージャ .....	16-6
INSTANCE プロシージャ .....	16-7
INTERVAL プロシージャ .....	16-7
BROKEN プロシージャ .....	16-8
RUN プロシージャ .....	16-9
USER_EXPORT プロシージャ .....	16-10
USER_EXPORT プロシージャ .....	16-10

## 17 DBMS\_LOB

要件 .....	17-2
LOB ロケータ .....	17-2
データ型 .....	17-3
定数 .....	17-3
例外 .....	17-4
セキュリティ .....	17-4
規則および制限事項 .....	17-5

BFILE 特有の規則および制限事項.....	17-6
テンポラリ LOB.....	17-9
テンポラリ LOB の使用上の注意.....	17-10
テンポラリ LOB の例外.....	17-12
<b>サブプログラムの要約.....</b>	<b>17-12</b>
APPEND プロシージャ.....	17-14
CLOSE プロシージャ.....	17-15
COMPARE ファンクション.....	17-16
COPY プロシージャ.....	17-19
CREATETEMPORARY プロシージャ.....	17-21
ERASE プロシージャ.....	17-22
FILECLOSE プロシージャ.....	17-24
FILECLOSEALL プロシージャ.....	17-25
FILEEXISTS ファンクション.....	17-26
FILEGETNAME プロシージャ.....	17-28
FILEISOPEN ファンクション.....	17-29
FILEOPEN プロシージャ.....	17-30
FREETEMPORARY プロシージャ.....	17-32
GETCHUNKSIZE ファンクション.....	17-33
GETLENGTH ファンクション.....	17-34
INSTR ファンクション.....	17-36
ISOPEN ファンクション.....	17-39
ISTEMPORARY ファンクション.....	17-40
LOADFROMFILE プロシージャ.....	17-40
OPEN プロシージャ.....	17-43
READ プロシージャ.....	17-44
SUBSTR ファンクション.....	17-47
TRIM プロシージャ.....	17-50
WRITE プロシージャ.....	17-52
WRITEAPPEND プロシージャ.....	17-54

## 18 DBMS\_LOCK

要件.....	18-2
セキュリティ.....	18-2
ロックの表示と監視.....	18-2

定数.....	18-2
<b>サブプログラムの要約</b> .....	18-4
ALLOCATE_UNIQUE プロシージャ .....	18-4
REQUEST ファンクション .....	18-6
CONVERT ファンクション .....	18-7
RELEASE ファンクション .....	18-9
SLEEP プロシージャ .....	18-10
例.....	18-10

## 19 DBMS\_LOGMNR

LogMiner の使用方法 .....	19-2
定数.....	19-2
ブレース・ホルダ列の使用法.....	19-3
<b>サブプログラムの要約</b> .....	19-5
ADD_LOGFILE プロシージャ .....	19-5
START_LOGMNR プロシージャ.....	19-6
END_LOGMNR プロシージャ .....	19-8
例.....	19-8

## 20 DBMS\_LOGMNR\_D

ディクショナリ・ファイルの作成.....	20-2
<b>サブプログラムの要約</b> .....	20-2
BUILD プロシージャ .....	20-2
例.....	20-3

## 21 DBMS\_OFFLINE\_OG

<b>サブプログラムの要約</b> .....	21-2
BEGIN_INSTANTIATION プロシージャ .....	21-2
BEGIN_LOAD プロシージャ .....	21-3
END_INSTANTIATION プロシージャ .....	21-4
END_LOAD プロシージャ .....	21-5
RESUME_SUBSET_OF_MASTERS プロシージャ .....	21-6

## 22 DBMS\_OFFLINE\_SNAPSHOT

サブプログラムの要約.....	22-2
BEGIN_LOAD プロシージャ .....	22-2
END_LOAD プロシージャ .....	22-3

## 23 DBMS\_OLAP

要件 .....	23-2
エラー・メッセージ .....	23-2
サブプログラムの要約.....	23-3
ESTIMATE_SUMMARY_SIZE プロシージャ .....	23-3
EVALUATE_UTILIZATION プロシージャ .....	23-4
EVALUATE_UTILIZATION_W プロシージャ .....	23-4
RECOMMEND_MV プロシージャ .....	23-5
RECOMMEND_MV_W プロシージャ .....	23-7
VALIDATE_DIMENSION プロシージャ .....	23-8
DBMS_OLAP インタフェース表 .....	23-9

## 24 DBMS\_ORACLE\_TRACE\_AGENT

セキュリティ .....	24-2
サブプログラムの要約.....	24-2
SET_ORACLE_TRACE_IN_SESSION プロシージャ .....	24-2
例 .....	24-3

## 25 DBMS\_ORACLE\_TRACE\_USER

サブプログラムの要約.....	25-2
SET_ORACLE_TRACE プロシージャ .....	25-2
例 .....	25-2

## 26 DBMS\_OUTPUT

セキュリティ .....	26-2
エラー .....	26-2
型 .....	26-2
DBMS_OUTPUT の使用方法 .....	26-2

<b>サブプログラムの要約</b> .....	26-3
ENABLE プロシージャ.....	26-3
DISABLE プロシージャ.....	26-4
PUT および PUT_LINE プロシージャ.....	26-4
NEW_LINE プロシージャ.....	26-6
GET_LINE および GET_LINES プロシージャ.....	26-6
例.....	26-8

## 27 DBMS\_PCLXUTIL

DBMS_PCLXUTIL の使用方法.....	27-2
制限事項.....	27-3
<b>サブプログラムの要約</b> .....	27-3
BUILD_PART_INDEX プロシージャ.....	27-3
例.....	27-4

## 28 DBMS\_PIPE

パブリック・パイプ.....	28-2
プライベート・パイプ.....	28-2
パイプの使用法.....	28-3
セキュリティ.....	28-3
定数.....	28-4
エラー.....	28-4
<b>サブプログラムの要約</b> .....	28-4
CREATE_PIPE ファンクション.....	28-5
PACK_MESSAGE プロシージャ.....	28-7
SEND_MESSAGE ファンクション.....	28-8
RECEIVE_MESSAGE ファンクション.....	28-10
NEXT_ITEM_TYPE ファンクション.....	28-12
UNPACK_MESSAGE プロシージャ.....	28-13
REMOVE_PIPE ファンクション.....	28-14
PURGE プロシージャ.....	28-15
RESET_BUFFER プロシージャ.....	28-16
UNIQUE_SESSION_NAME ファンクション.....	28-16
例.....	28-17

## 29 DBMS\_PROFILER

DBMS_PROFILER の使用方法 .....	29-2
収集されたデータ .....	29-3
要件 .....	29-3
エラー・コード .....	29-3
<b>サブプログラムの要約</b> .....	29-4
START_PROFILER ファンクション .....	29-4
STOP_PROFILER ファンクション .....	29-5
FLUSH_DATA ファンクション .....	29-5
GET_VERSION プロシージャ .....	29-6
INTERNAL_VERSION_CHECK ファンクション .....	29-6

## 30 DBMS\_RANDOM

要件 .....	30-2
<b>サブプログラムの要約</b> .....	30-2
INITIALIZE プロシージャ .....	30-2
SEED プロシージャ .....	30-3
RANDOM ファンクション .....	30-3
TERMINATE プロシージャ .....	30-3

## 31 DBMS\_RECTIFIER\_DIFF

<b>サブプログラムの要約</b> .....	31-2
DIFFERENCES プロシージャ .....	31-2
RECTIFY プロシージャ .....	31-5

## 32 DBMS\_REFRESH

<b>サブプログラムの要約</b> .....	32-2
ADD プロシージャ .....	32-2
CHANGE プロシージャ .....	32-3
DESTROY プロシージャ .....	32-5
MAKE プロシージャ .....	32-6
REFRESH プロシージャ .....	32-8

SUBTRACT プロシージャ .....	32-9
-----------------------	------

## 索引

はじめに .....	xxxi
------------	------

### 33 DBMS\_REPAIR

セキュリティ .....	33-2
列挙型 .....	33-2
例外 .....	33-2
<b>サブプログラムの要約</b> .....	33-4
ADMIN_TABLES プロシージャ .....	33-4
CHECK_OBJECT プロシージャ .....	33-5
DUMP_ORPHAN_KEYS プロシージャ .....	33-7
FIX_CORRUPT_BLOCKS プロシージャ .....	33-9
REBUILD_FREELISTS プロシージャ .....	33-10
SKIP_CORRUPT_BLOCKS プロシージャ .....	33-11

### 34 DBMS\_REPCAT

<b>サブプログラムの要約</b> .....	34-2
ADD_GROUPED_COLUMN プロシージャ .....	34-5
ADD_MASTER_DATABASE プロシージャ .....	34-7
ADD_PRIORITY_datatype プロシージャ .....	34-8
ADD_SITE_PRIORITY_SITE プロシージャ .....	34-9
ADD_conflicttype_RESOLUTION プロシージャ .....	34-10
ALTER_MASTER_PROPAGATION プロシージャ .....	34-14
ALTER_MASTER_REPOBJECT プロシージャ .....	34-15
ALTER_PRIORITY プロシージャ .....	34-17
ALTER_PRIORITY_datatype プロシージャ .....	34-18
ALTER_SITE_PRIORITY プロシージャ .....	34-19
ALTER_SITE_PRIORITY_SITE プロシージャ .....	34-20
ALTER_SNAPSHOT_PROPAGATION プロシージャ .....	34-21
CANCEL_STATISTICS プロシージャ .....	34-22
COMMENT_ON_COLUMN_GROUP プロシージャ .....	34-23



COMMENT_ON_PRIORITY_GROUP/PRIORITY プロシージャ .....	34-24
COMMENT_ON_REPGROUP プロシージャ.....	34-25
COMMENT_ON_REPSITES プロシージャ.....	34-26
COMMENT_ON_REPOBJECT プロシージャ .....	34-27
COMMENT_ON_conflicttype_RESOLUTION プロシージャ.....	34-28
COMPARE_OLD_VALUES プロシージャ.....	34-29
CREATE_MASTER_REPGROUP プロシージャ .....	34-31
CREATE_MASTER_REPOBJECT プロシージャ .....	34-32
CREATE_SNAPSHOT_REPGROUP プロシージャ .....	34-35
CREATE_SNAPSHOT_REPOBJECT プロシージャ.....	34-36
DEFINE_COLUMN_GROUP プロシージャ.....	34-38
DEFINE_PRIORITY_GROUP プロシージャ.....	34-39
DEFINE_SITE_PRIORITY プロシージャ.....	34-41
DO_DEFERRED_REPCAT_ADMIN プロシージャ.....	34-41
DROP_COLUMN_GROUP プロシージャ .....	34-42
DROP_GROUPED_COLUMN プロシージャ.....	34-43
DROP_MASTER_REPGROUP プロシージャ.....	34-44
DROP_MASTER_REPOBJECT プロシージャ.....	34-46
DROP_PRIORITY プロシージャ.....	34-47
DROP_PRIORITY_GROUP プロシージャ .....	34-47
DROP_PRIORITY_datatype プロシージャ .....	34-48
DROP_SITE_PRIORITY プロシージャ .....	34-50
DROP_SITE_PRIORITY_SITE プロシージャ .....	34-50
DROP_SNAPSHOT_REPGROUP プロシージャ.....	34-51
DROP_SNAPSHOT_REPOBJECT プロシージャ .....	34-52
DROP_conflicttype_RESOLUTION プロシージャ .....	34-53
EXECUTE_DDL プロシージャ.....	34-55
GENERATE_REPLICATION_SUPPORT プロシージャ .....	34-56
GENERATE_SNAPSHOT_SUPPORT プロシージャ.....	34-57
MAKE_COLUMN_GROUP プロシージャ .....	34-59
PURGE_MASTER_LOG プロシージャ.....	34-60
PURGE_STATISTICS プロシージャ.....	34-61
REFRESH_SNAPSHOT_REPGROUP プロシージャ .....	34-62
REGISTER_SNAPSHOT_REPGROUP プロシージャ.....	34-63
REGISTER_STATISTICS プロシージャ .....	34-64

RELOCATE_MASTERDEF プロシージャ .....	34-65
REMOVE_MASTER_DATABASES プロシージャ .....	34-66
REPCAT_IMPORT_CHECK プロシージャ .....	34-67
RESUME_MASTER_ACTIVITY プロシージャ .....	34-68
SEND_OLD_VALUES プロシージャ .....	34-69
SET_COLUMNS プロシージャ .....	34-71
SUSPEND_MASTER_ACTIVITY プロシージャ .....	34-72
SWITCH_SNAPSHOT_MASTER プロシージャ .....	34-73
UNREGISTER_SNAPSHOT_REPGROUP プロシージャ .....	34-74
VALIDATE ファンクション .....	34-75
WAIT_MASTER_LOG プロシージャ .....	34-77

## 35 DBMS\_REPCAT\_ADMIN

サブプログラムの要約 .....	35-2
GRANT_ADMIN_ANY_SCHEMA プロシージャ .....	35-2
GRANT_ADMIN_SCHEMA プロシージャ .....	35-3
REGISTER_USER_REPGROUP プロシージャ .....	35-3
REVOKE_ADMIN_ANY_SCHEMA プロシージャ .....	35-5
REVOKE_ADMIN_SCHEMA プロシージャ .....	35-6
UNREGISTER_USER_REPGROUP プロシージャ .....	35-6

## 36 DBMS\_REPCAT\_INSTANTIATE

サブプログラムの要約 .....	36-2
DROP_SITE_INSTANTIATION プロシージャ .....	36-2
INSTANTIATE_OFFLINE ファンクション .....	36-3
INSTANTIATE_ONLINE ファンクション .....	36-5

## 37 DBMS\_REPCAT\_RGT

サブプログラムの要約 .....	37-2
ALTER_REFRESH_TEMPLATE プロシージャ .....	37-4
ALTER_TEMPLATE_OBJECT プロシージャ .....	37-6
ALTER_TEMPLATE_PARM プロシージャ .....	37-9
ALTER_USER_AUTHORIZATION プロシージャ .....	37-10
ALTER_USER_PARM_VALUE プロシージャ .....	37-11

COMPARE_TEMPLATES ファンクション .....	37-14
COPY_TEMPLATE ファンクション .....	37-15
CREATE_OBJECT_FROM_EXISTING ファンクション .....	37-17
CREATE_REFRESH_TEMPLATE ファンクション .....	37-18
CREATE_TEMPLATE_OBJECT ファンクション .....	37-20
CREATE_TEMPLATE_PARM ファンクション .....	37-22
CREATE_USER_AUTHORIZATION ファンクション .....	37-24
CREATE_USER_PARM_VALUE ファンクション .....	37-26
DELETE_RUNTIME_PARMS プロシージャ .....	37-28
DROP_ALL_OBJECTS プロシージャ .....	37-28
DROP_ALL_TEMPLATE_PARMS プロシージャ .....	37-29
DROP_ALL_TEMPLATE_SITES プロシージャ .....	37-30
DROP_ALL_TEMPLATES プロシージャ .....	37-31
DROP_ALL_USER_AUTHORIZATIONS プロシージャ .....	37-31
DROP_ALL_USER_PARM_VALUES プロシージャ .....	37-32
DROP_REFRESH_TEMPLATE プロシージャ .....	37-33
DROP_SITE_INSTANTIATION プロシージャ .....	37-34
DROP_TEMPLATE_OBJECT プロシージャ .....	37-35
DROP_TEMPLATE_PARM プロシージャ .....	37-36
DROP_USER_AUTHORIZATION プロシージャ .....	37-36
DROP_USER_PARM_VALUE プロシージャ .....	37-37
GET_RUNTIME_PARM_ID ファンクション .....	37-38
INSERT_RUNTIME_PARMS プロシージャ .....	37-39
INstantiate_OFFLINE ファンクション .....	37-40
INstantiate_ONLINE ファンクション .....	37-42
LOCK_TEMPLATE_EXCLUSIVE プロシージャ .....	37-44
LOCK_TEMPLATE_SHARED プロシージャ .....	37-45

## 38 DBMS\_REPUTIL

サブプログラムの要約 .....	38-2
REPLICATION_OFF プロシージャ .....	38-2
REPLICATION_ON プロシージャ .....	38-2
REPLICATION_IS_ON ファンクション .....	38-3
FROM_REMOTE ファンクション .....	38-3
GLOBAL_NAME ファンクション .....	38-3

## 39 DBMS\_RESOURCE\_MANAGER

要件 .....	39-2
<b>サブプログラムの要約</b> .....	39-2
CREATE_PLAN プロシージャ .....	39-3
UPDATE_PLAN プロシージャ .....	39-4
DELETE_PLAN プロシージャ .....	39-4
DELETE_PLAN_CASCADE プロシージャ .....	39-5
CREATE_CONSUMER_GROUP プロシージャ .....	39-6
UPDATE_CONSUMER_GROUP プロシージャ .....	39-6
DELETE_CONSUMER_GROUP プロシージャ .....	39-7
CREATE_PLAN_DIRECTIVE プロシージャ .....	39-8
UPDATE_PLAN_DIRECTIVE プロシージャ .....	39-9
DELETE_PLAN_DIRECTIVE プロシージャ .....	39-10
CREATE_PENDING_AREA プロシージャ .....	39-10
VALIDATE_PENDING_AREA プロシージャ .....	39-12
CLEAR_PENDING_AREA プロシージャ .....	39-12
SUBMIT_PENDING_AREA プロシージャ .....	39-12
SET_INITIAL_CONSUMER_GROUP プロシージャ .....	39-16
SWITCH_CONSUMER_GROUP_FOR_SESS プロシージャ .....	39-17
SWITCH_CONSUMER_GROUP_FOR_USER プロシージャ .....	39-17

## 40 DBMS\_RESOURCE\_MANAGER\_PRIVS

<b>サブプログラムの要約</b> .....	40-2
GRANT_SYSTEM_PRIVILEGE プロシージャ .....	40-2
REVOKE_SYSTEM_PRIVILEGE プロシージャ .....	40-3
GRANT_SWITCH_CONSUMER_GROUP プロシージャ .....	40-3
REVOKE_SWITCH_CONSUMER_GROUP プロシージャ .....	40-5

## 41 DBMS\_RLS

動的な述語 .....	41-2
セキュリティ .....	41-3
使用上の注意 .....	41-3
<b>サブプログラムの要約</b> .....	41-3
ADD_POLICY プロシージャ .....	41-3

DROP_POLICY プロシージャ .....	41-5
REFRESH_POLICY プロシージャ .....	41-6
ENABLE_POLICY プロシージャ .....	41-7
例 .....	41-8

## 42 DBMS\_ROWID

使用上の注意 .....	42-2
要件 .....	42-2
ROWID のタイプ .....	42-3
例外 .....	42-4
<b>サブプログラムの要約</b> .....	42-4
ROWID_CREATE ファンクション .....	42-5
ROWID_INFO プロシージャ .....	42-6
ROWID_TYPE ファンクション .....	42-7
ROWID_OBJECT ファンクション .....	42-8
ROWID_RELATIVE_FNO ファンクション .....	42-9
ROWID_BLOCK_NUMBER ファンクション .....	42-10
ROWID_ROW_NUMBER ファンクション .....	42-10
ROWID_TO_ABSOLUTE_FNO ファンクション .....	42-11
ROWID_TO_EXTENDED ファンクション .....	42-12
ROWID_TO_RESTRICTED ファンクション .....	42-14
ROWID_VERIFY ファンクション .....	42-14

## 43 DBMS\_SESSION

要件 .....	43-2
<b>サブプログラムの要約</b> .....	43-2
SET_ROLE プロシージャ .....	43-3
SET_SQL_TRACE プロシージャ .....	43-3
SET_NLS プロシージャ .....	43-4
CLOSE_DATABASE_LINK プロシージャ .....	43-4
RESET_PACKAGE プロシージャ .....	43-5
UNIQUE_SESSION_ID ファンクション .....	43-6
IS_ROLE_ENABLED ファンクション .....	43-7
IS_SESSION_ALIVE ファンクション .....	43-7

SET_CLOSE_CACHED_OPEN_CURSORS プロシージャ .....	43-8
FREE_UNUSED_USER_MEMORY プロシージャ .....	43-8
SET_CONTEXT プロシージャ .....	43-11
LIST_CONTEXT プロシージャ .....	43-11
SWITCH_CURRENT_CONSUMER_GROUP プロシージャ .....	43-12
例 .....	43-14

## 44 DBMS\_SHARED\_POOL

インストール時の注意 .....	44-2
使用上の注意 .....	44-2
<b>サブプログラムの要約</b> .....	44-2
SIZES プロシージャ .....	44-2
KEEP プロシージャ .....	44-3
UNKEEP プロシージャ .....	44-5
ABORTED_REQUEST_THRESHOLD プロシージャ .....	44-5

## 45 DBMS\_SNAPSHOT

<b>サブプログラムの要約</b> .....	45-2
BEGIN_TABLE_REORGANIZATION プロシージャ .....	45-2
END_TABLE_REORGANIZATION プロシージャ .....	45-3
I_AM_A_REFRESH ファンクション .....	45-3
PURGE_DIRECT_LOAD_LOG プロシージャ .....	45-4
PURGE_LOG プロシージャ .....	45-4
PURGE_SNAPSHOT_FROM_LOG プロシージャ .....	45-5
REFRESH プロシージャ .....	45-7
REFRESH_ALL_MVIEWS プロシージャ .....	45-9
REFRESH_DEPENDENT プロシージャ .....	45-10
REGISTER_SNAPSHOT プロシージャ .....	45-12
UNREGISTER_SNAPSHOT プロシージャ .....	45-13

## 46 DBMS\_SPACE

セキュリティ .....	46-2
要件 .....	46-2
<b>サブプログラムの要約</b> .....	46-2

UNUSED_SPACE プロシージャ .....	46-2
FREE_BLOCKS プロシージャ .....	46-3
例.....	46-5

## 47 DBMS\_SPACE\_ADMIN

セキュリティ .....	47-2
定数.....	47-2
<b>サブプログラムの要約</b> .....	47-3
SEGMENT_VERIFY プロシージャ .....	47-3
SEGMENT_CORRUPT プロシージャ .....	47-4
SEGMENT_DROP_CORRUPT プロシージャ.....	47-5
SEGMENT_DUMP プロシージャ .....	47-6
TABLESPACE_VERIFY プロシージャ .....	47-7
TABLESPACE_FIX_BITMAPS プロシージャ .....	47-7
TABLESPACE_REBUILD_BITMAPS プロシージャ.....	47-8
TABLESPACE_REBUILD_QUOTAS プロシージャ .....	47-9
TABLESPACE_MIGRATE_FROM_LOCAL プロシージャ .....	47-10
例.....	47-10

## 48 DBMS\_SQL

DBMS_SQL の使用方法.....	48-2
定数.....	48-3
型.....	48-3
例外 .....	48-3
実行フロー .....	48-4
セキュリティ .....	48-7
<b>サブプログラムの要約</b> .....	48-8
OPEN_CURSOR ファンクション .....	48-9
PARSE プロシージャ .....	48-10
BIND_VARIABLE プロシージャ .....	48-12
BIND_ARRAY プロシージャ .....	48-12
問合せの処理.....	48-16
DEFINE_COLUMN プロシージャ .....	48-17
DEFINE_ARRAY プロシージャ.....	48-18

DEFINE_COLUMN_LONG プロシージャ .....	48-20
EXECUTE ファンクション .....	48-20
EXECUTE_AND_FETCH ファンクション .....	48-21
FETCH_ROWS ファンクション .....	48-21
COLUMN_VALUE プロシージャ .....	48-22
COLUMN_VALUE_LONG プロシージャ .....	48-24
VARIABLE_VALUE プロシージャ .....	48-25
更新、挿入、削除の処理 .....	48-27
IS_OPEN ファンクション .....	48-27
DESCRIBE_COLUMNS プロシージャ .....	48-28
CLOSE_CURSOR プロシージャ .....	48-30
エラーの位置 .....	48-31
LAST_ERROR_POSITION ファンクション .....	48-31
LAST_ROW_COUNT ファンクション .....	48-31
LAST_ROW_ID ファンクション .....	48-32
LAST_SQL_FUNCTION_CODE ファンクション .....	48-32
例 .....	48-33

## 49 DBMS\_STATS

DBMS_STATS の使用方法 .....	49-2
型 .....	49-2
<b>サブプログラムの要約</b> .....	49-3
統計情報の設定または取得 .....	49-5
PREPARE_COLUMN_VALUES プロシージャ .....	49-5
SET_COLUMN_STATS プロシージャ .....	49-7
SET_INDEX_STATS プロシージャ .....	49-9
SET_TABLE_STATS プロシージャ .....	49-10
CONVERT_RAW_VALUE プロシージャ .....	49-11
GET_COLUMN_STATS プロシージャ .....	49-13
GET_INDEX_STATS プロシージャ .....	49-14
GET_TABLE_STATS プロシージャ .....	49-15
DELETE_COLUMN_STATS プロシージャ .....	49-16
DELETE_INDEX_STATS プロシージャ .....	49-17
DELETE_TABLE_STATS プロシージャ .....	49-18
DELETE_SCHEMA_STATS プロシージャ .....	49-20



DELETE_DATABASE_STATS プロシージャ .....	49-20
統計情報の転送.....	49-21
CREATE_STAT_TABLE プロシージャ .....	49-22
DROP_STAT_TABLE プロシージャ .....	49-22
EXPORT_COLUMN_STATS プロシージャ .....	49-23
EXPORT_INDEX_STATS プロシージャ .....	49-24
EXPORT_TABLE_STATS プロシージャ .....	49-25
EXPORT_SCHEMA_STATS プロシージャ .....	49-26
EXPORT_DATABASE_STATS プロシージャ .....	49-26
IMPORT_COLUMN_STATS プロシージャ .....	49-27
IMPORT_INDEX_STATS プロシージャ .....	49-28
IMPORT_TABLE_STATS プロシージャ .....	49-29
IMPORT_SCHEMA_STATS プロシージャ .....	49-30
IMPORT_DATABASE_STATS プロシージャ .....	49-30
オブティマイザ統計情報の収集.....	49-31
GATHER_INDEX_STATS プロシージャ.....	49-31
GATHER_TABLE_STATS プロシージャ.....	49-32
GATHER_SCHEMA_STATS プロシージャ.....	49-34
GATHER_DATABASE_STATS プロシージャ.....	49-37
GENERATE_STATS プロシージャ.....	49-39
例.....	49-40

## 50 DBMS\_TRACE

要件.....	50-2
制限事項.....	50-2
定数.....	50-2
DBMS_TRACE の使用方法.....	50-2
<b>サブプログラムの要約</b> .....	50-4
SET_PLSQL_TRACE プロシージャ.....	50-4
CLEAR_PLSQL_TRACE プロシージャ .....	50-4
PLSQL_TRACE_VERSION プロシージャ .....	50-5

## 51 DBMS\_TRANSACTION

要件.....	51-2
---------	------

<b>サブプログラムの要約</b> .....	51-2
READ_ONLY プロシージャ .....	51-3
READ_WRITE プロシージャ .....	51-3
ADVISE_ROLLBACK プロシージャ .....	51-3
ADVISE_NOTHING プロシージャ .....	51-4
ADVISE_COMMIT プロシージャ .....	51-4
USE_ROLLBACK_SEGMENT プロシージャ .....	51-4
COMMIT_COMMENT プロシージャ .....	51-5
COMMIT_FORCE プロシージャ .....	51-5
COMMIT プロシージャ .....	51-6
SAVEPOINT プロシージャ .....	51-6
ROLLBACK プロシージャ .....	51-7
ROLLBACK_SAVEPOINT プロシージャ .....	51-7
ROLLBACK_FORCE プロシージャ .....	51-8
BEGIN_DISCRETE_TRANSACTION プロシージャ .....	51-8
PURGE_MIXED プロシージャ .....	51-9
PURGE_LOST_DB_ENTRY プロシージャ .....	51-10
LOCAL_TRANSACTION_ID ファンクション .....	51-12
STEP_ID ファンクション .....	51-12

## 52 DBMS\_TTS

例外 .....	52-2
<b>サブプログラムの要約</b> .....	52-2
TRANSPORT_SET_CHECK プロシージャ .....	52-2
DOWNGRADE プロシージャ .....	52-3

## 53 DBMS\_UTILITY

要件 .....	53-2
型 .....	53-2
<b>サブプログラムの要約</b> .....	53-2
COMPILE_SCHEMA プロシージャ .....	53-4
ANALYZE_SCHEMA プロシージャ .....	53-5
ANALYZE_DATABASE プロシージャ .....	53-6
FORMAT_ERROR_STACK ファンクション .....	53-7

FORMAT_CALL_STACK ファンクション .....	53-7
IS_PARALLEL_SERVER ファンクション .....	53-8
GET_TIME ファンクション .....	53-8
GET_PARAMETER_VALUE ファンクション .....	53-9
NAME_RESOLVE プロシージャ .....	53-10
NAME_TOKENIZE プロシージャ .....	53-12
COMMA_TO_TABLE プロシージャ .....	53-12
TABLE_TO_COMMA プロシージャ .....	53-13
PORT_STRING ファンクション .....	53-14
DB_VERSION プロシージャ .....	53-14
MAKE_DATA_BLOCK_ADDRESS ファンクション .....	53-15
DATA_BLOCK_ADDRESS_FILE ファンクション .....	53-15
DATA_BLOCK_ADDRESS_BLOCK ファンクション .....	53-16
GET_HASH_VALUE ファンクション .....	53-17
ANALYZE_PART_OBJECT プロシージャ .....	53-18
EXEC_DDL_STATEMENT プロシージャ .....	53-19
CURRENT_INSTANCE ファンクション .....	53-19
ACTIVE_INSTANCES プロシージャ .....	53-19

## 54 DEBUG\_EXTPROC

要件 .....	54-2
インストレーション時の注意 .....	54-2
DEBUG_EXTPROC の使用方法 .....	54-2
<b>サブプログラムの要約</b> .....	54-3
STARTUP_EXTPROC_AGENT プロシージャ .....	54-3

## 55 OUTLN\_PKG

要件 .....	55-2
セキュリティ .....	55-2
<b>サブプログラムの要約</b> .....	55-2
DROP_UNUSED プロシージャ .....	55-2
DROP_BY_CAT プロシージャ .....	55-3
UPDATE_BY_CAT プロシージャ .....	55-3

## 56 UTL\_COLL

サブプログラムの要約.....	56-2
IS_LOCATOR ファンクション .....	56-2
例.....	56-3

## 57 UTL\_FILE

セキュリティ .....	57-2
ファイルの所有権と保護.....	57-3
例 ( UNIX 固有 ).....	57-3
型.....	57-4
例外 .....	57-4
サブプログラムの要約.....	57-5
FOPEN ファンクション .....	57-6
IS_OPEN ファンクション .....	57-7
FCLOSE プロシージャ.....	57-8
FCLOSE_ALL プロシージャ.....	57-8
GET_LINE プロシージャ .....	57-9
PUT プロシージャ.....	57-10
NEW_LINE プロシージャ .....	57-11
PUT_LINE プロシージャ .....	57-12
PUTF プロシージャ.....	57-12
FFLUSH プロシージャ .....	57-14
FOPEN ファンクション .....	57-14

## 58 UTL\_HTTP

例外 .....	58-2
使用上の注意.....	58-3
サブプログラムの要約.....	58-3
REQUEST ファンクション .....	58-3
REQUEST_PIECES ファンクション.....	58-5
例.....	58-6

## 59 UTL\_RAW

使用上の注意.....	59-2
-------------	------

<b>サブプログラムの要約</b> .....	59-2
CONCAT ファンクション .....	59-3
CAST_TO_RAW ファンクション .....	59-4
CAST_TO_VARCHAR2 ファンクション .....	59-5
LENGTH ファンクション .....	59-6
SUBSTR ファンクション .....	59-7
TRANSLATE ファンクション .....	59-8
TRANSLITERATE ファンクション .....	59-10
OVERLAY ファンクション .....	59-11
COPIES ファンクション .....	59-13
XRANGE ファンクション .....	59-14
REVERSE ファンクション .....	59-15
COMPARE ファンクション .....	59-16
CONVERT ファンクション .....	59-17
BIT_AND ファンクション .....	59-19
BIT_OR ファンクション .....	59-20
BIT_XOR ファンクション .....	59-21
BIT_COMPLEMENT ファンクション .....	59-22

## 60 UTL\_REF

要件 .....	60-2
データ型 .....	60-2
例外 .....	60-2
セキュリティ .....	60-3
<b>サブプログラムの要約</b> .....	60-4
SELECT_OBJECT プロシージャ .....	60-4
LOCK_OBJECT プロシージャ .....	60-5
UPDATE_OBJECT プロシージャ .....	60-6
DELETE_OBJECT プロシージャ .....	60-7
例 .....	60-8

## 索引



---

# はじめに

このマニュアルでは、Oracle8i、リリース 8.1.5 プログラムに付属の Oracle パッケージについて説明します。このマニュアルの情報は、すべてのプラットフォームで実行される Oracle Server の各バージョンに対して適用され、システム固有の情報は含まれていません。

この章では、次の項目について説明します。

- [パッケージの概要](#)
- [対象読者](#)
- [関連マニュアル](#)
- [表記規則](#)

## パッケージの概要

パッケージとは、論理的に関連している PL/SQL の型、アイテムおよびサブプログラムをグループ化するスキーマ・オブジェクトです。パッケージには通常、仕様部と本体の 2 つの部分があります（本体は必要ない場合もあります）。仕様部はアプリケーションへのインタフェースで、使用可能な型、変数、定数、例外、カーソルおよびサブプログラムを宣言します。本体はカーソルとサブプログラムを完全に定義し、仕様部を完全にインプリメントします。

パッケージは、サブプログラムとは異なり、コールしたり、パラメータ化したり、ネスト化することはできません。ただし、パッケージの書式は、サブプログラムと類似しています。

```
CREATE PACKAGE name AS -- specification (visible part)
    -- public type and item declarations
    -- subprogram specifications
END [name];
```

```
CREATE PACKAGE BODY name AS -- body (hidden part)
    -- private type and item declarations
    -- subprogram bodies
[BEGIN
    -- initialization statements]
END [name];
```

仕様部は、アプリケーションで参照できるパブリック宣言を含んでいます。本体は、インプリメンテーションの詳細とプライベート宣言を含んでいます。これらはアプリケーションからは参照できません。

パッケージ本体へのインタフェース（パッケージ仕様部）を変更せずに、パッケージ本体をデバッグ、拡張および置換できます。

パッケージを作成し、Oracle データベースに格納するためには、CREATE PACKAGE および CREATE PACKAGE BODY 文を使用します。これらの文は、SQL\*Plus または Enterprise Manager から対話形式で実行できます。

アプリケーションから参照およびアクセスできるのは、パッケージ仕様部の宣言部のみです。パッケージ本体にあるインプリメンテーションの詳細は参照およびアクセスできません。このため、本体（インプリメンテーション）は、コール側のプログラムを再コンパイルせずに変更できます。

パッケージを使用する利点には、モジュール性、容易なアプリケーション設計、情報の非公開、機能性の追加およびパフォーマンスの向上などがあります。



## 対象読者

このマニュアルは、Oracle パッケージを使用する、または Oracle パッケージを使用する予定があるすべてのユーザーを対象にしています。また、システム・アナリスト、プロジェクト・マネージャ、およびデータベース・アプリケーションの開発やチューニングを行う予定があるユーザーにも有効です。

このマニュアルでは、読者がアプリケーション・プログラミングに関する作業知識があり、リレーショナル・データベース・システムの情報にアクセスするための構造化問合せ言語 (SQL) の使用経験が十分にあることを前提としています。

一部の項では、オブジェクト指向プログラミングの基本概念を理解していることも前提としています。

## 関連マニュアル

詳細は、Oracle8i マニュアル・セットにある次のマニュアルを参照してください。

- 『Oracle8i アプリケーション開発者ガイド 基礎編』
- 『PL/SQL ユーザーズ・ガイドおよびリファレンス』

# 表記規則

このマニュアルでは次の表記規則を使用します。

表記	意味
...	省略符号が文またはコマンド内で使用されている場合は、例に直接関係のない部分が省略されていることを示します。
英大文字のテキスト	パッケージ名、コマンド・キーワード、データベース・オブジェクト名およびファイル名などを強調するために使用します。
コード例	SQL、Oracle Enterprise Manager 行モード (Server Manager) および SQL*Plus のコマンドまたは文は、クーリエ・フォントで表示されます。 例： <pre>INSERT INTO emp (empno, ename) VALUES (1000, 'SMITH'); ALTER TABLESPACE users ADD DATAFILE 'users2.ora' SIZE 50K;</pre>
< >	山カッコは、ユーザーが指定する必要のある名前を示します。
[ ]	大カッコは選択が任意の項目を示します。選択肢の中から 1 つ選択するか、または何も入力しなくてもかまいません。
\$	ドル記号は、OpenVMS の DIGITAL CommandLanguage プロンプトおよび Digital UNIX の Bourne シェル・プロンプトを表します。

---

## DBMS\_REPAIR

DBMS\_REPAIR には、データ破損修復プロシージャが含まれており、ユーザーは表と索引にある破損ブロックを検出して修復できます。可能な場合は破損をつき止め、再構築中または修復中にオブジェクトの使用を続行することができます。

---

**注意：** DBMS\_REPAIR パッケージは、データベース管理者のみの使用を目的としています。アプリケーション開発者を対象にした機能ではありません。

---

**関連項目：** DBMS\_REPAIR パッケージの使用方法的詳細は、『Oracle8i 管理者ガイド』を参照してください。

# セキュリティ

このパッケージの所有者は SYS です。他のユーザーに実行権限は付与されていません。

# 列挙型

DBMS\_REPAIR パッケージは、パラメータ値の指定に使用するいくつかの列挙定数を定義します。列挙定数にはパッケージ名を接頭辞として付加する必要があります。たとえば、DBMS\_REPAIR.TABLE\_OBJECT と記述します。

表 33-1 は、パラメータと列挙定数の一覧です。

表 33-1 DBMS\_REPAIR の列挙型

パラメータ	定数
object_type	TABLE_OBJECT, INDEX_OBJECT, CLUSTER_OBJECT
action	CREATE_ACTION, DROP_ACTION, PURGE_ACTION
table_type	REPAIR_TABLE, ORPHAN_TABLE
flags	SKIP_FLAG, NOSKIP_FLAG

注意： デフォルトの table\_name は、table\_type が REPAIR\_TABLE のときは REPAIR\_TABLE で、table\_type が ORPHAN\_TABLE のときは ORPHAN\_KEY\_TABLE です。

# 例外

表 33-2 DBMS\_REPAIR の例外

例外	説明	アクション
942	指定した表が存在していないと、DROP_ACTION 時に DBMS_REPAIR.ADMIN_TABLES によって戻されます。	
955	指定した表がすでに存在していると、DBMS_REPAIR.CREATE_ACTION に よって戻されます。	
24120	指定した DBMS_REPAIR プロシージャ に無効なパラメータが渡されました。	有効なパラメータ値を指定するか、またはパラメータのデフォルトを使用してください。

**表 33-2 DBMS\_REPAIR の例外**

例外	説明	アクション
24122	ブロック範囲の指定に誤りがあります。	BLOCK_START と BLOCK_END パラメータに正しい値を指定してください。
24123	指定した機能を使用しようとしたが、その機能はまだインプリメントされていません。	この機能は使用しないでください。
24124	ACTION パラメータに無効な値が指定されました。	ACTION パラメータには、CREATE_ACTION、PURGE_ACTION または DROP_ACTION のいずれかを指定してください。
24125	DBMS_REPAIR.CHECK_OBJECT の実行後に削除または切り捨てられたオブジェクトの破損ブロックを修正しようとした。	DBMS_REPAIR.ADMIN_TABLES で修復表をパージし、DBMS_REPAIR.CHECK_OBJECT を実行して、修正対象の破損ブロックがあるかどうかを確認してください。
24127	CREATE_ACTION 以外の ACTION で TABLESPACE パラメータが指定されました。	CREATE_ACTION 以外のアクションの実行時には、TABLESPACE は指定しないでください。
24128	パーティション化されていないオブジェクトに対して、パーティション名が指定されました。	パーティション名は、オブジェクトがパーティション化されているときのみ指定してください。
24129	接頭辞を指定しないで、table_name パラメータを渡そうとした。	有効な table_name パラメータを渡してください。
24130	存在しない修復表または親なし表を指定しようとした。	table_name パラメータに有効な値を指定してください。
24131	誤った定義内容の修復表または親なし表を指定しようとした。	正しく作成された表を参照する表名を指定してください。
24132	30 文字を超える表名を指定しようとした。	table_name パラメータに有効な値を指定してください。

# サブプログラムの要約

表 33-3 DBMS\_REPAIR パッケージのサブプログラム

サブプログラム	説明
<a href="#">ADMIN_TABLES プロシージャ</a> ( 33-4 ページ )	DBMS_REPAIR パッケージの修復表と親なしキー表に対して、作成、パージおよび削除処理を実行する管理ファンクションを提供します。
<a href="#">CHECK_OBJECT プロシージャ</a> ( 33-5 ページ )	表または索引の破損を検出し、レポートします。
<a href="#">DUMP_ORPHAN_KEYS プロシージャ</a> ( 33-7 ページ )	破損データ・ブロック内の行を指す索引エントリをレポートします。
<a href="#">FIX_CORRUPT_BLOCKS プロシージャ</a> ( 33-9 ページ )	CHECK_OBJECT によって破損が検出されたブロックにソフトウェア破損のマークを付けます。
<a href="#">REBUILD_FREELISTS プロシージャ</a> ( 33-10 ページ )	オブジェクトの空きリストを再作成します。
<a href="#">SKIP_CORRUPT_BLOCKS プロシージャ</a> ( 33-11 ページ )	表と索引のスキャン時に破損マークのあるブロックを無視するか、または破損マークのあるブロックが検出された場合にORA-1578 をレポートするかを設定します。

## ADMIN\_TABLES プロシージャ

このプロシージャは、DBMS\_REPAIR パッケージの修復表と親なしキー表に対する管理ファンクションを提供します。

### 構文

```
DBMS_REPAIR.ADMIN_TABLES (  
    table_name IN      VARCHAR2,  
    table_type IN      BINARY_INTEGER,  
    action      IN      BINARY_INTEGER,  
    tablespace IN      VARCHAR2          DEFAULT NULL);
```

パラメータ

表 33-4 ADMIN\_TABLES プロシージャのパラメータ

パラメータ	説明
table_name	処理する表の名前。指定した table_type に基づいて、ORPHAN_KEY_TABLE または REPAIR_TABLE をデフォルト設定します。指定するときは、適切な接頭辞 ( ORPHAN_ または REPAIR_ ) が表名に必要です。
table_type	表の型。ORPHAN_TABLE または REPAIR_TABLE のいずれかです。  「 <a href="#">列挙型</a> 」( 33-2 ページ ) を参照してください。
action	実行する管理アクションを示します。  CREATE_ACTION、PURGE_ACTION または DROP_ACTION のいずれかです。CREATE_ACTION の指定時に、表がすでに存在しているとエラーが戻されます。PURGE_ACTION を指定すると、存在しないオブジェクトに関連付けられている表内の行はすべて削除されます。DROP_ACTION の指定時に、表が存在していないとエラーが戻されます。  CREATE_ACTION と DROP_ACTION を指定すると、DBA_<table_name> という名前の関連ビューが、それぞれ作成され、削除されます。このビューは、存在しないオブジェクトに関連付けられている行を排除するように定義されています。  SYS スキーマ内に作成されます。  「 <a href="#">列挙型</a> 」( 33-2 ページ ) を参照してください。
tablespace	表の作成時に使用する表領域を示します。  デフォルトでは、SYS のデフォルト表領域が使用されます。CREATE_ACTION 以外のときに表領域を指定するとエラーが戻されます。

CHECK\_OBJECT プロシージャ

このプロシージャは、指定したオブジェクトをチェックし、破損と修復指示に関する情報を修復表に移入します。

妥当性チェックでは、オブジェクト内のすべてのブロックがチェックされます。オブジェクトの一部をチェック対象とする場合は、オプションで、DBA 範囲、パーティション名またはサブパーティション名を指定することもできます。

構文

```
DBMS_REPAIR.CHECK_OBJECT (
    schema_name      IN  VARCHAR2,
    object_name      IN  VARCHAR2,
    partition_name   IN  VARCHAR2      DEFAULT NULL,
    object_type      IN  BINARY_INTEGER DEFAULT TABLE_OBJECT,
    repair_table_name IN  VARCHAR2      DEFAULT 'REPAIR_TABLE',
    flags            IN  BINARY_INTEGER DEFAULT NULL,
    relative_fno     IN  BINARY_INTEGER DEFAULT NULL,
    block_start      IN  BINARY_INTEGER DEFAULT NULL,
    block_end        IN  BINARY_INTEGER DEFAULT NULL,
    corrupt_count    OUT BINARY_INTEGER);
```

パラメータ

表 33-5 CHECK\_OBJECT プロシージャのパラメータ

パラメータ	説明
schema_name	チェックするオブジェクトのスキーマ名。
object_name	チェックする表または索引の名前。
partition_name	チェックするパーティションまたはサブパーティションの名前。 パーティション・オブジェクトで、partition_name が指定されていない場合は、すべてのパーティションとサブパーティションがチェックされます。パーティション・オブジェクトで、指定したパーティションにサブパーティションが含まれている場合は、すべてのサブパーティションがチェックされます。
object_type	処理するオブジェクトの型。TABLE_OBJECT (デフォルト) または INDEX_OBJECT のいずれかです。 <a href="#">「列挙型」</a> ( 33-2 ページ ) を参照してください。
repair_table_name	情報を移入する修復表の名前。 この表は、SYS スキーマに存在している必要があります。修復表を作成するには、admin_tables プロシージャを使用します。デフォルト名は REPAIR_TABLE です。
flags	将来使用のために確保。
relative_fno	相対ファイル番号。ブロック範囲の指定時に使用します。
block_start	ブロック範囲を指定する場合に、最初に処理するブロックを指定します。オブジェクトが単一表、パーティションまたはサブパーティションの場合のみ指定できます。



表 33-5 CHECK\_OBJECT プロシージャのパラメータ

パラメータ	説明
block_end	ブロック範囲を指定する場合に、最後に処理するブロックを指定します。オブジェクトが単一表、パーティションまたはサブパーティションの場合のみ指定できます。block_start または block_end のいずれか一方のみ指定した場合、他方の値は、ファイル内の第 1 ブロックまたは最終ブロックにそれぞれデフォルト設定されます。
corrupt_count	レポートされた破損数。

## DUMP\_ORPHAN\_KEYS プロシージャ

このプロシージャは、破損データ・ブロック内の行を指す索引エントリをレポートします。検出された該当索引エントリごとに、指定した親なし表に行が挿入されます。

修復表が指定されている場合は、ソフトウェア破損のマークがあるすべてのデータ・ブロックの他に、実表に関連付けられている破損ブロックが処理されます。修復表が指定されていない場合は、破損マークのあるブロックのみ処理されます。

この情報は、表内で失われた行を再構築する場合や診断の目的に使用されます。

## 構文

```
DBMS_REPAIR.DUMP_ORPHAN_KEYS (
    schema_name      IN  VARCHAR2,
    object_name       IN  VARCHAR2,
    partition_name    IN  VARCHAR2      DEFAULT NULL,
    object_type       IN  BINARY_INTEGER DEFAULT INDEX_OBJECT,
    repair_table_name IN  VARCHAR2      DEFAULT 'REPAIR_TABLE',
    orphan_table_name IN  VARCHAR2      DEFAULT 'ORPHAN_KEYS_TABLE',
    flags             IN  BINARY_INTEGER DEFAULT NULL,
    key_count         OUT BINARY_INTEGER);
```

## パラメータ

表 33-6 DUMP\_ORPHAN\_KEYS プロシージャのパラメータ

パラメータ	説明
schema_name	スキーマ名。
object_name	オブジェクト名。

表 33-6 DUMP\_ORPHAN\_KEYS プロシージャのパラメータ

パラメータ	説明
partition_name	処理するパーティションまたはサブパーティションの名前。  パーティション・オブジェクトの場合に、partition_name を指定しない場合は、すべてのパーティションとサブパーティションが処理されます。パーティション・オブジェクトの場合に、指定したパーティションにサブパーティションが含まれている場合は、すべてのサブパーティションが処理されます。
object_type	処理するオブジェクトの型。デフォルトは INDEX_OBJECT です。  「 <a href="#">列挙型</a> 」( 33-2 ページ ) を参照してください。
repair_table_name	実表の破損ブロックに関する情報を含んだ修復表の名前。  指定した表は、SYS スキーマに存在している必要があります。表を作成するには、admin_tables プロシージャを使用します。
orphan_table_name	破損データ・ブロック内の行を参照する各索引エントリに関する情報を移入する親なしキー表の名前。  指定した表は、SYS スキーマに存在している必要があります。表を作成するには、admin_tables プロシージャを使用します。
flags	将来使用のために確保。
key_count	処理された索引エントリ数。

## FIX\_CORRUPT\_BLOCKS プロシージャ

このプロシージャは、`check_object` プロシージャによって事前に生成された修復表の情報に基づいて、指定したオブジェクト内の破損ブロックを修正します。

ブロックに変更を加える前に、そのブロックがまだ破損状態であることを確認するチェックが行われます。破損ブロックは、そのブロックにソフトウェア破損のマークを付けることによって修復されます。修復が有効になると、修復表内の関連行が修正タイムスタンプで更新されます。

### 構文

```
DBMS_REPAIR.FIX_CORRUPT_BLOCKS (  
    schema_name      IN  VARCHAR2,  
    object_name       IN  VARCHAR2,  
    partition_name    IN  VARCHAR2      DEFAULT NULL,  
    object_type        IN  BINARY_INTEGER DEFAULT TABLE_OBJECT,  
    repair_table_name  IN  VARCHAR2      DEFAULT 'REPAIR_TABLE',  
    flags              IN  BINARY_INTEGER DEFAULT NULL,  
    fix_count          OUT BINARY_INTEGER);
```

### パラメータ

表 33-7 FIX\_CORRUPT\_BLOCKS プロシージャのパラメータ

パラメータ	説明
<code>schema_name</code>	スキーマ名。
<code>object_name</code>	修正対象の破損ブロックがあるオブジェクトの名前。
<code>partition_name</code>	処理するパーティションまたはサブパーティションの名前。  パーティション・オブジェクトの場合に、 <code>partition_name</code> を指定しない場合は、すべてのパーティションとサブパーティションが処理されます。パーティション・オブジェクトの場合に、指定したパーティションにサブパーティションが含まれている場合は、すべてのサブパーティションが処理されます。
<code>object_type</code>	処理するオブジェクトの型。TABLE_OBJECT (デフォルト) または INDEX_OBJECT のいずれかです。  <a href="#">「列挙型」</a> ( 33-2 ページ ) を参照してください。
<code>repair_table_name</code>	修復指示を含んだ修復表の名前。  SYS スキーマに存在している必要があります。
<code>flags</code>	将来使用のために確保。
<code>fix_count</code>	修正されたブロック数。

## REBUILD\_FREELISTS プロシージャ

このプロシージャは、指定したオブジェクトの空きリストを再作成します。すべての空きブロックは、マスター空きリストに格納されます。その他の空きリストはすべてゼロになります。

オブジェクトに複数の空きリスト・グループがある場合、空きブロックは、ラウンドロビン方式で異なるグループに割り当てられ、すべての空きリスト間に配分されます。

### 構文

```
DBMS_REPAIR.REBUILD_FREELISTS (  
    schema_name      IN VARCHAR2,  
    partition_name   IN VARCHAR2      DEFAULT NULL,  
    object_type       IN BINARY_INTEGER DEFAULT TABLE_OBJECT);
```

### パラメータ

表 33-8 REBUILD\_FREELISTS プロシージャのパラメータ

パラメータ	説明
schema_name	スキーマ名。
object_name	空きリストを再作成するオブジェクトの名前。
partition_name	空きリストを再作成するパーティションまたはサブパーティションの名前。  パーティション・オブジェクトの場合に partition_name を指定しない場合は、すべてのパーティションとサブパーティションが処理されます。パーティション・オブジェクトので、指定したパーティションにサブパーティションが含まれている場合は、すべてのサブパーティションが処理されます。
object_type	処理するオブジェクトの型。TABLE_OBJECT (デフォルト) または INDEX_OBJECT のいずれかです。  <a href="#">「列挙型」</a> ( 33-2 ページ ) を参照してください。

# SKIP\_CORRUPT\_BLOCKS プロシージャ

このプロシージャは、指定したオブジェクトの索引と表のスキャン時に、破損ブロックのスキップを使用可能または使用禁止にします。

オブジェクトが表のときは、スキップが表とその索引に適用されます。オブジェクトがクラスタのときは、クラスタ内のすべての表とその各索引に適用されます。

## 構文

```
DBMS_REPAIR.SKIP_CORRUPT_BLOCKS (  
    schema_name IN VARCHAR2,  
    object_name IN VARCHAR2,  
    object_type IN BINARY_INTEGER DEFAULT TABLE_OBJECT,  
    flags        IN BINARY_INTEGER DEFAULT SKIP_FLAG);
```

## パラメータ

表 33-9 SKIP\_CORRUPT\_BLOCKS プロシージャのパラメータ

パラメータ	説明
schema_name	処理するオブジェクトのスキーマ名。
object_name	オブジェクトの名前。
partition_name (optional)	処理するパーティションまたはサブパーティションの名前。  パーティション・オブジェクトで、partition_name を指定しない場合は、すべてのパーティションとサブパーティションが処理されます。パーティション・オブジェクトで、指定したパーティションにサブパーティションが含まれている場合は、すべてのサブパーティションが処理されます。
object_type	処理するオブジェクトの型。TABLE_OBJECT (デフォルト) または CLUSTER_OBJECT のいずれかです。  「 <a href="#">列挙型</a> 」( 33-2 ページ ) を参照してください。
flags	SKIP_FLAG を指定すると、索引と表のスキャン時に、そのオブジェクトのソフトウェア破損ブロックのスキップがオンになります。NOSKIP_FLAG を指定すると、ソフトウェア破損ブロックが検出されたときに、ORA-1578 エラーが戻されます。  「 <a href="#">列挙型</a> 」( 33-2 ページ ) を参照してください。



---

## DBMS\_REPCAT

DBMS\_REPCAT は、レプリケーション・カタログとレプリケーション環境を管理および更新するためのルーチンを提供します。

# サブプログラムの要約

表 34-1 DBMS\_REPCAT パッケージのサブプログラム

サブプログラム	説明
<a href="#">ADD_GROUPED_COLUMN プロシージャ</a> (34-5 ページ)	既存の列グループにメンバーを追加します。
<a href="#">ADD_MASTER_DATABASE プロシージャ</a> (34-7 ページ)	レプリケート環境に別のマスター・サイトを追加します。
<a href="#">ADD_PRIORITY_datatype プロシージャ</a> (34-8 ページ)	優先順位グループにメンバーを追加します。
<a href="#">ADD_SITE_PRIORITY_SITE プロシージャ</a> (34-9 ページ)	サイト優先順位グループに新規サイトを追加します。
<a href="#">ADD_conflicttype_RESOLUTION プロシージャ</a> (34-10 ページ)	更新、削除または一意性の競合の解消方法を指定します。
<a href="#">ALTER_MASTER_PROPAGATION プロシージャ</a> (34-14 ページ)	指定したマスター・サイトで指定したオブジェクト・グループの伝播方法を変更します。
<a href="#">ALTER_MASTER_REPOBJECT プロシージャ</a> (34-15 ページ)	レプリケート環境内のオブジェクトを変更します。
<a href="#">ALTER_PRIORITY プロシージャ</a> (34-17 ページ)	指定した優先順位グループ・メンバーに関連付けられている優先順位レベルを変更します。
<a href="#">ALTER_PRIORITY_datatype プロシージャ</a> (34-18 ページ)	優先順位グループ内のメンバーの値を変更します。
<a href="#">ALTER_SITE_PRIORITY プロシージャ</a> (34-19 ページ)	指定したサイトに関連付けられている優先順位レベルを変更します。
<a href="#">ALTER_SITE_PRIORITY_SITE プロシージャ</a> (34-20 ページ)	指定した優先順位レベルに関連付けられているサイトを変更します。
<a href="#">ALTER_SNAPSHOT_PROPAGATION プロシージャ</a> (34-21 ページ)	現行スナップショット・サイトで指定したオブジェクト・グループの伝播方法を変更します。
<a href="#">CANCEL_STATISTICS プロシージャ</a> (34-22 ページ)	表の更新競合、一意性競合および削除競合の正常な解消に関する統計の収集を停止します。
<a href="#">COMMENT_ON_COLUMN_GROUP プロシージャ</a> (34-23 ページ)	列グループの RepColumn_Group ビューのコメント・フィールドを更新します。
<a href="#">COMMENT_ON_PRIORITY_GROUP/PRIORITY プロシージャ</a> (34-24 ページ)	(サイト) 優先順位グループの REPPRIORITY_GROUP ビューのコメント・フィールドを更新します。



表 34-1 DBMS\_REPCAT パッケージのサブプログラム

サブプログラム	説明
<a href="#">COMMENT_ON_REPGROUP プロシージャ</a> (34-25 ページ)	レプリケート・オブジェクト・グループの REPGROUP ビューのコメント・フィールドを更新します。
<a href="#">COMMENT_ON_REPSITES プロシージャ</a> (34-26 ページ)	レプリケート・サイトの RepSite ビューのコメント・フィールドを更新します。
<a href="#">COMMENT_ON_REPOBJECT プロシージャ</a> (34-27 ページ)	レプリケート・オブジェクトの RepObject ビューのコメント・フィールドを更新します。
<a href="#">COMMENT_ON_conflicttype_RESOLUTION プロシージャ</a> (34-28 ページ)	競合解消ルーチンの RepResolution ビューのコメント・フィールドを更新します。
<a href="#">COMPARE_OLD_VALUES プロシージャ</a> (34-29 ページ)	マスター・サイトとスナップショット・サイトでレプリケート表の元の列を比較します。
<a href="#">CREATE_MASTER_REPGROUP プロシージャ</a> (34-31 ページ)	新規に、空で休止状態のマスター・レプリケーション・オブジェクト・グループを作成します。
<a href="#">CREATE_MASTER_REPOBJECT プロシージャ</a> (34-32 ページ)	オブジェクトをレプリケート・オブジェクトとして定義します。
<a href="#">CREATE_SNAPSHOT_REPGROUP プロシージャ</a> (34-35 ページ)	ローカル・データベース内に、新規で空のスナップショット・レプリケーション・オブジェクト・グループを作成します。
<a href="#">CREATE_SNAPSHOT_REPOBJECT プロシージャ</a> (34-36 ページ)	スナップショット・サイトにレプリケート・オブジェクトを追加します。
<a href="#">DEFINE_COLUMN_GROUP プロシージャ</a> (34-38 ページ)	空の列グループを作成します。
<a href="#">DEFINE_PRIORITY_GROUP プロシージャ</a> (34-39 ページ)	レプリケート・オブジェクト・グループに新規の優先順位グループを作成します。
<a href="#">DEFINE_SITE_PRIORITY プロシージャ</a> (34-41 ページ)	レプリケート・オブジェクト・グループに新規のサイト優先順位グループを作成します。
<a href="#">DO_DEFERRED_REPCAT_ADMIN プロシージャ</a> (34-41 ページ)	現行マスター・サイトで指定したレプリケート・オブジェクト・グループ、またはすべてのマスター・サイトに対する未処理のローカル遅延管理プロシージャを実行します。
<a href="#">DROP_COLUMN_GROUP プロシージャ</a> (34-42 ページ)	列グループを削除します。
<a href="#">DROP_GROUPED_COLUMN プロシージャ</a> (34-43 ページ)	列グループからメンバーを削除します。
<a href="#">DROP_MASTER_REPGROUP プロシージャ</a> (34-44 ページ)	現行サイトからレプリケート・オブジェクト・グループを削除します。

表 34-1 DBMS\_REPCAT パッケージのサブプログラム

サブプログラム	説明
<a href="#">DROP_MASTER_REPOBJECT プロシージャ</a> (34-46 ページ)	レプリケート・オブジェクト・グループからレプリケート・オブジェクトを削除します。
<a href="#">DROP_PRIORITY プロシージャ</a> (34-47 ページ)	優先順位レベルに従って優先順位グループのメンバーを削除します。
<a href="#">DROP_PRIORITY_GROUP プロシージャ</a> (34-47 ページ)	指定したレプリケート・オブジェクト・グループの優先順位グループを削除します。
<a href="#">DROP_PRIORITY_datatype プロシージャ</a> (34-48 ページ)	値による優先順位グループのメンバーを削除します。
<a href="#">DROP_SITE_PRIORITY プロシージャ</a> (34-50 ページ)	指定したレプリケート・オブジェクト・グループのサイト優先順位グループを削除します。
<a href="#">DROP_SITE_PRIORITY_SITE プロシージャ</a> (34-50 ページ)	サイト優先順位グループから、名前によって指定したサイトを削除します。
<a href="#">DROP_SNAPSHOT_REPGROUP プロシージャ</a> (34-51 ページ)	レプリケート環境からスナップショット・サイトを削除します。
<a href="#">DROP_SNAPSHOT_REPOBJECT プロシージャ</a> (34-52 ページ)	スナップショット・サイトからレプリケート・オブジェクトを削除します。
<a href="#">DROP_conflicttype_RESOLUTION プロシージャ</a> (34-53 ページ)	更新、削除または一意性競合解消ルーチンを削除します。
<a href="#">EXECUTE_DDL プロシージャ</a> (34-55 ページ)	各マスター・サイトで実行する DDL を提供します。
<a href="#">GENERATE_REPLICATION_SUPPORT プロシージャ</a> (34-56 ページ)	レプリケーションのサポートに必要なトリガー、パッケージおよびプロシージャを生成します。
<a href="#">GENERATE_SNAPSHOT_SUPPORT プロシージャ</a> (34-57 ページ)	トリガーを起動し、更新可能スナップショットのレプリケーションまたはプロシージャ・レプリケーションのサポートに必要なパッケージを生成します。
<a href="#">MAKE_COLUMN_GROUP プロシージャ</a> (34-59 ページ)	1 つ以上のメンバーを含んだ新しい列グループを作成します。
<a href="#">PURGE_MASTER_LOG プロシージャ</a> (34-60 ページ)	指定した識別番号、発信元またはレプリケート・オブジェクト・グループに対応する RepCatLog 内のローカル・メッセージを削除します。
<a href="#">PURGE_STATISTICS プロシージャ</a> (34-61 ページ)	RepResolution_Statistics ビューから情報を削除します。
<a href="#">REFRESH_SNAPSHOT_REPGROUP プロシージャ</a> (34-62 ページ)	スナップショット・サイトのオブジェクト・グループを、関連するマスター・サイトから最新データでリフレッシュします。

表 34-1 DBMS\_REPCAT パッケージのサブプログラム

サブプログラム	説明
<a href="#">REGISTER_SNAPSHOT_REPGROUP プロシージャ</a> (34-63 ページ)	repcat_repsite の挿入、変更および削除を行うことによって、それぞれのマスター・サイトのスナップショット管理を容易にします。
<a href="#">REGISTER_STATISTICS プロシージャ</a> (34-64 ページ)	表の更新競合、削除競合および一意性競合の正常な解消に関する情報を収集します。
<a href="#">RELOCATE_MASTERDEF プロシージャ</a> (34-65 ページ)	マスター定義サイトをレプリケート環境内の別のマスター・サイトに変更します。
<a href="#">REMOVE_MASTER_DATABASES プロシージャ</a> (34-66 ページ)	レプリケート環境から 1 つ以上のマスター・データベースを削除します。
<a href="#">REPCAT_IMPORT_CHECK プロシージャ</a> (34-67 ページ)	レプリケート・オブジェクトまたはアドバンスド・レプリケーション機能で使用するオブジェクトのエクスポートまたはインポートの実行後、レプリケート・オブジェクト・グループ内のオブジェクトの識別子と状態値が適切かどうかを確認します。
<a href="#">RESUME_MASTER_ACTIVITY プロシージャ</a> (34-68 ページ)	レプリケート環境の休止後、通常のレプリケーション・アクティビティを再開します。
<a href="#">SEND_OLD_VALUES プロシージャ</a> (34-69 ページ)	マスター・サイトとスナップショット・サイトでレプリケート表の元の列を送信します。
<a href="#">SET_COLUMNS プロシージャ</a> (34-71 ページ)	行レベル・レプリケーションの使用時に、表のどの列を比較するかを判断します。
<a href="#">SUSPEND_MASTER_ACTIVITY プロシージャ</a> (34-72 ページ)	オブジェクト・グループのレプリケーション・アクティビティを中断します。
<a href="#">SWITCH_SNAPSHOT_MASTER プロシージャ</a> (34-73 ページ)	スナップショット・レプリケート・オブジェクトのマスター・データベースを別のマスター・サイトに変更します。
<a href="#">UNREGISTER_SNAPSHOT_REPGROUP プロシージャ</a> (34-74 ページ)	repcat\$_repsite の挿入、変更および削除を行うことによって、それぞれのマスター・サイトのスナップショット管理を容易にします。
<a href="#">VALIDATE ファンクション</a> (34-75 ページ)	複数マスター・レプリケーション環境のキー状態が正しいかどうかを検証します。
<a href="#">WAIT_MASTER_LOG プロシージャ</a> (34-77 ページ)	マスター・サイトに非同期で伝播された変更内容が適用されたかどうかを判断します。

## ADD\_GROUPED\_COLUMN プロシージャ

このプロシージャは、既存の列グループにメンバーを追加します。このプロシージャは、マスター定義サイトからコールする必要があります。

構文

```
DBMS_REPCAT.ADD_GROUPED_COLUMN (  
    sname                IN    VARCHAR2,  
    oname                IN    VARCHAR2,  
    column_group         IN    VARCHAR2,  
    list_of_column_names IN    VARCHAR2 | DBMS_REPCAT.VARCHAR2S);
```

パラメータ

表 34-2 ADD\_GROUPED\_COLUMN プロシージャのパラメータ

パラメータ	説明
sname	レプリケート表が置かれているスキーマ。
oname	列グループが関連付けられているレプリケート表の名前。
column_group	メンバーを追加する列グループの名前。
list_of_column_names	指定した列グループに追加する列の名前。列名は、名前のカンマで区切られたリストまたは PL/SQL 表のいずれかで示します。PL/SQL 表は dbms_repcat.varchar2 型にしてください。表内のすべての列を含んだ列グループを作成するには、単一の値 '*' を使用します。

例外

表 34-3 ADD\_GROUPED\_COLUMN プロシージャの例外

例外	説明
nonmasterdef	起動サイトがマスター定義サイトではありません。
missingobject	指定した表が存在しません。
missinggroup	指定した列グループが存在しません。
missingcolumn	指定した列が、指定した表内に存在していません。
duplicatecolumn	指定した列は、すでに別の列グループのメンバーです。
missingschema	指定したスキーマが存在しません。
notquiesced	指定した表が属するオブジェクト・グループが休止状態ではありません。

## ADD\_MASTER\_DATABASE プロシージャ

このプロシージャは、レプリケート環境に別のマスター・サイトを追加します。また、すべてのトリガーと関連するパッケージを既存のマスター・サイトで再生成します。このプロシージャは、マスター定義サイトからコールする必要があります。

### 構文

```
DBMS_REPCAT.ADD_MASTER_DATABASE (  
    gname                IN    VARCHAR2,  
    master               IN    VARCHAR2,  
    use_existing_objects IN    BOOLEAN := TRUE,  
    copy_rows            IN    BOOLEAN := TRUE,  
    comment              IN    VARCHAR2 := '',  
    propagation_mode     IN    VARCHAR2 := 'ASYNCHRONOUS',  
    fname               IN    VARCHAR2 := NULL);
```

### パラメータ

表 34-4 ADD\_MASTER\_DATABASE プロシージャのパラメータ

パラメータ	説明
gname	レプリケートするオブジェクト・グループの名前。このオブジェクト・グループは、マスター定義サイトにすでに存在している必要があります。
master	新規マスター・データベースの完全修飾データベース名。
use_existing_objects	スキーマ内にすでに存在するオブジェクトと同じ型、同じ形式のオブジェクトを、新規マスター・サイトで再使用する場合は、TRUE を指定します。これらの変更内容の適用方法の詳細は、『Oracle8i レプリケーション・ガイド』の「マルチマスタ・レプリケーションの使用方法」を参照してください。
copy_rows	新規マスター・サイトの表の初期の内容を、マスター定義サイトの表の内容と一致させる場合は TRUE を指定します。
comment	RepSites ビューの MASTER_COMMENT フィールドに追加されるコメント。
propagation_mode	新規マスター・データベースとの間の変更内容の送受信方法を示します。指定できる値は SYNCHRONOUS と ASYNCHRONOUS です。
fname	内部使用のためのシステム・パラメータ。オラクル社カスタマ・サポートから指示がない限り、このパラメータは設定しないでください。

例外

表 34-5 ADD\_MASTER\_DATABASE プロシージャの例外

例外	説明
nonmasterdef	起動サイトがマスター定義サイトではありません。
notquiesced	レプリケート・オブジェクト・グループが中断されていません。
missingrepgroup	オブジェクト・グループが、指定したデータベース・サイトに存在していません。
commfailure	新規マスターにアクセスできません。
typefailure	伝播モードの指定に誤りがあります。
notcompat	互換モードは 7.3.0.0 以上である必要があります。
duplrepgrp	マスター・サイトがすでに存在します。

ADD\_PRIORITY\_datatype プロシージャ

このプロシージャは、優先順位グループにメンバーを追加します。このプロシージャは、マスター定義サイトからコールする必要があります。コールする必要があるプロシージャは、`priority` 列のデータ型によって決まります。このプロシージャは、`priority` 列の有効な値ごとに 1 回ずつコールしてください。

**関連項目：** 詳細は、『Oracle8i レプリケーション・ガイド』の「競合の解消」を参照してください。

構文

```
DBMS_REPCAT.ADD_PRIORITY_datatype (  
  gname          IN   VARCHAR2,  
  pgroup         IN   VARCHAR2,  
  value          IN   datatype,  
  priority       IN   NUMBER);
```

`datatype` には次のいずれかを指定します。

```
{ NUMBER  
| VARCHAR2  
| CHAR  
| DATE  
| RAW  
| NCHAR  
| NVARCHAR2 }
```

## パラメータ

表 34-6 ADD\_PRIORITY\_datatype プロシージャのパラメータ

パラメータ	説明
gname	優先順位グループを作成するレプリケート・オブジェクト・グループ。
pgroup	優先順位グループの名前。
value	優先順位グループ・メンバーの値。この値は、この優先順位グループを使用している表の priority 列に関連した値となる可能性があります。
priority	この値の優先順位。数値が大きいほど優先順位は高くなります。

## 例外

表 34-7 ADD\_PRIORITY\_datatype プロシージャの例外

例外	説明
nomasterdef	起動サイトがマスター定義サイトではありません。
duplicatevalue	指定した値が優先順位グループ内にすでに存在しています。
duplicatepriority	指定した優先順位が優先順位グループ内にすでに存在しています。
missingrepgroup	指定したレプリケート・オブジェクト・グループが存在しません。
missingprioritygroup	指定した優先順位グループが存在しません。
typefailure	優先順位グループに対して指定した値のデータ型が正しくありません。
notquiesced	指定したレプリケート・オブジェクト・グループが休止状態ではありません。

## ADD\_SITE\_PRIORITY\_SITE プロシージャ

このプロシージャは、サイト優先順位グループに新規サイトを追加します。このプロシージャは、マスター定義サイトからコールする必要があります。

**関連項目：** 詳細は、『Oracle8i レプリケーション・ガイド』の「競合の解消」を参照してください。

構文

```
DBMS_REPCAT.ADD_SITE_PRIORITY_SITE (  
    gname          IN   VARCHAR2,  
    name           IN   VARCHAR2,  
    site           IN   VARCHAR2,  
    priority       IN   NUMBER);
```

パラメータ

表 34-8 ADD\_SITE\_PRIORITY\_SITE プロシージャのパラメータ

パラメータ	説明
gname	サイトをグループに追加するレプリケート・オブジェクト・グループ。
name	メンバーを追加するサイト優先順位グループの名前。
site	追加するサイトのグローバル・データベース名。
priority	追加するサイトの優先順位レベル。数値が大きいほど優先順位レベルは高くなります。

例外

表 34-9 ADD\_SITE\_PRIORITY\_SITE プロシージャの例外

例外	説明
nonmasterdef	起動サイトがマスター定義サイトではありません。
missingrepgroup	指定したレプリケート・オブジェクト・グループが存在しません。
missingpriority	指定したサイト優先順位グループが存在しません。
duplicatepriority	指定した優先順位レベルが、グループ内の別のサイト用にすでに存在しています。
duplicatevalue	指定したサイトが、サイト優先順位グループ内にすでに存在しています。
notquiesced	レプリケート・オブジェクト・グループが休止状態ではありません。

ADD\_conflicttype\_RESOLUTION プロシージャ

このプロシージャは、更新、削除または一意性の競合の解消方法を指定します。このプロシージャは、マスター定義サイトからコールする必要があります。コールする必要があるプロシージャは、そのルーチンが解消する競合の型によって決まります。



競合の型	プロシージャ名
更新	ADD_UPDATE_RESOLUTION
一意性	ADD_UNIQUE_RESOLUTION
削除	ADD_DELETE_RESOLUTION

**関連項目：** 更新競合の解消方法の指定、一意性競合の解消方法の選択、および削除競合の解消方法の割当てに関する詳細は、『Oracle8i レプリケーション・ガイド』の「競合の解消」を参照してください。

## 構文

```
DBMS_REPCAT.ADD_UPDATE_RESOLUTION (
    sname           IN    VARCHAR2,
    oname           IN    VARCHAR2,
    column_group    IN    VARCHAR2,
    sequence_no     IN    NUMBER,
    method          IN    VARCHAR2,
    parameter_column_name IN VARCHAR2 | DBMS_REPCAT.VARCHAR2S,
    priority_group  IN    VARCHAR2      := NULL,
    function_name   IN    VARCHAR2      := NULL,
    comment         IN    VARCHAR2      := NULL);
```

```
DBMS_REPCAT.ADD_DELETE_RESOLUTION (
    sname           IN    VARCHAR2,
    oname           IN    VARCHAR2,
    sequence_no     IN    NUMBER,
    parameter_column_name IN VARCHAR2 | DBMS_REPCAT.VARCHAR2S,
    function_name   IN    VARCHAR2,
    comment         IN    VARCHAR2      := NULL,
    method          IN    VARCHAR2      := 'USER FUNCTION');
```

```
DBMS_REPCAT.ADD_UNIQUE_RESOLUTION(
    sname           IN    VARCHAR2,
    oname           IN    VARCHAR2,
    constraint_name  IN    VARCHAR2,
    sequence_no     IN    NUMBER,
    method          IN    VARCHAR2,
    parameter_column_name IN VARCHAR2 | DBMS_REPCAT.VARCHAR2S,
    function_name   IN    VARCHAR2      := NULL,
    comment         IN    VARCHAR2      := NULL);
```

パラメータ

表 34-10 ADD\_conflicttype\_RESOLUTION プロシージャのパラメータ

パラメータ	説明
sname	レプリケートする表が含まれているスキーマの名前。
oname	競合解消ルーチンを追加する表の名前。
column_group	競合解消ルーチンを追加する列グループの名前。更新競合解消ルーチンのみが列グループを必要とします。
constraint_name	競合解消ルーチンを追加する一意制約または一意索引の名前。一意索引の名前が関連する一意制約の名前と異なる場合は、一意索引の名前を使用します。一意性競合解消ルーチンのみが制約名を必要とします。
sequence_no	指定した競合解消方法を適用する順序。
method	<p>作成する競合解消ルーチンの型。アドバンスト・レプリケーションで提供されている標準ルーチンのいずれかの名前を指定できます。または、独自のルーチンを作成している場合は、USER FUNCTION を選択し、FUNCTION_NAME 引数にそのルーチン名を指定してください。</p> <p>このリリースでサポートされている方法は、MINIMUM、MAXIMUM、LATEST TIMESTAMP、EARLIEST TIMESTAMP、ADDITIVE、AVERAGE、PRIORITY GROUP、SITE PRIORITY、OVERWRITE、DISCARD ( 以上更新競合の場合 ) および APPEND SITE NAME、APPEND SEQUENCE NUMBER、DISCARD ( 以上一意性競合の場合 ) です。削除競合の標準ルーチンは提供されていません。</p>
parameter_column_name	<p>競合の解消に使用する列の名前。標準の方法では、単一の列で操作が行われます。たとえば、列グループに対して LATEST TIMESTAMP 方法を使用している場合は、この引数としてタイムスタンプを含んだ列の名前を渡す必要があります。USER FUNCTION を使用している場合は、任意の数の列を使用して競合を解消できます。</p> <p>この引数は、列名のカンマで区切られたリスト、または型 dbms_repcat.varchar2 の PL/SQL 表のいずれかを受け入れます。値 '*' は、競合の解消に表内のすべての列 ( または更新競合の場合は列グループ ) を使用することを意味します。 '*' を指定すると、列はユーザー・ファンクションにアルファベット順で渡されます。</p>
priority_group	<p>PRIORITY GROUP または SITE PRIORITY の更新競合解消方法を使用している場合は、自分で作成した優先順位グループの名前を指定する必要があります。</p> <p>『Oracle8i レプリケーション・ガイド』の「競合の解消」を参照してください。その他の方法を使用している場合は、この引数のデフォルト値 NULL を使用できます。この引数は更新競合の場合のみ適用できます。</p>

表 34-10 ADD\_conflicttype\_RESOLUTION プロシージャのパラメータ

パラメータ	説明
function_name	USER FUNCTION 方法を選択した場合、または削除競合解消ルーチンを追加する場合は、自分で作成した競合解消ルーチンの名前を指定する必要があります。標準方法の 1 つを使用している場合は、この引数のデフォルト値 NULL を使用できます。
comment	RepResolution ビューに追加されるユーザー・コメント。

## 例外

表 34-11 ADD\_conflicttype\_RESOLUTION プロシージャの例外

例外	説明
nonmasterdef	起動サイトがマスター定義サイトではありません。
missingobject	指定したオブジェクトは、行レベル・レプリケーションを使用する表として指定のスキーマ内に存在していません。
missingschema	指定したスキーマが存在しません。
missingcolumn	PARAMETER_COLUMN_NAME 引数の一部としてユーザーが指定した列が存在しません。
missinggroup	指定した列グループが存在しません。
missingprioritygroup	ユーザーが指定した優先順位グループがその表に存在していません。
invalidmethod	ユーザーが指定した解消方法は認識されていません。
invalidparameter	PARAMETER_COLUMN_NAME 引数に指定した列の数が正しくありません。(標準ルーチンは列名を 1 つしか使用しません。)
missingfunction	ユーザーが指定したユーザー・ファンクションが存在しません。
missingconstraint	一意性競合に対してユーザーが指定した制約が存在していません。
notquiesced	指定した表が属するオブジェクト・グループが休止状態ではありません。
duplicateresolution	指定した競合解消方法は、すでに登録されています。
paramtype	型が、優先順位グループに割り当てられた型と異なります。

## ALTER\_MASTER\_PROPAGATION プロシージャ

このプロシージャは、指定したマスター・サイトで指定したオブジェクト・グループの伝播方法を変更します。オブジェクト・グループは休止状態にしておいてください。このプロシージャは、マスター定義サイトからコールする必要があります。マスターが dblink\_list または dblink\_table に含まれている場合、そのデータベース・リンクは ALTER\_MASTER\_PROPAGATION によって無視されます。マスターから同じマスター自身への伝播モードは変更できません。

### 構文

```
DBMS_REPCAT.ALTER_MASTER_PROPAGATION (  
  gname                IN   VARCHAR2,  
  master               IN   VARCHAR2,  
  { dblink_list        IN   VARCHAR2  
  | dblink_table       IN   dbms_utility.dblink_array,}  
  propagation_mode     IN   VARCHAR2 := 'asynchronous',  
  comment              IN   VARCHAR2 := '' );
```

---

**注意：** このプロシージャはオーバーロードされています。dblink\_list パラメータと dblink\_table パラメータは、両方同時には指定できません。

---

### パラメータ

表 34-12 ALTER\_MASTER\_PROPAGATION プロシージャのパラメータ

パラメータ	説明
gname	伝播モードを変更するオブジェクト・グループの名前。
master	伝播モードを変更するマスター・サイトの名前。
dblink_list	伝播を変更するデータベース・リンクのカンマで区切られたリスト。NULL の場合は、変更対象のマスター・サイト以外のすべてのマスターがデフォルトとして使用されます。
dblink_table	伝播を変更するデータベース・リンクの PL/SQL 表（位置 1 から索引付け）。
propagation_mode	指定したマスター・サイトの変更内容を、データベース・リンクのリストで指定したサイトに伝播する方法を指定します。指定できる値は SYNCHRONOUS と ASYNCHRONOUS です。
comment	RepProp ビューに追加されるコメント。

例外

表 34-13 ALTER\_MASTER\_PROPAGATION プロシージャの例外

例外	説明
nonmasterdef	ローカル・サイトがマスター定義サイトではありません。
notquiesced	ローカル・サイトが休止状態ではありません。
typefailure	指定した伝播モードは認識されませんでした。
nonmaster	データベース・リンクのリストに、マスター・サイト以外のサイトが含まれています。

ALTER\_MASTER\_REPOBJECT プロシージャ

このプロシージャは、レプリケート環境内のオブジェクトを変更します。このプロシージャは、マスター定義サイトからコールする必要があります。

構文

```
DBMS_REPCAT.ALTER_MASTER_REPOBJECT (  
    sname          IN   VARCHAR2,  
    oname          IN   VARCHAR2,  
    type           IN   VARCHAR2,  
    ddl_text       IN   VARCHAR2,  
    comment        IN   VARCHAR2      := ' ',  
    retry          IN   BOOLEAN       := FALSE);
```

パラメータ

表 34-14 ALTER\_MASTER\_REPOBJECT プロシージャのパラメータ

パラメータ	説明
sname	変更するオブジェクトを含んだスキーマ。
oname	変更するオブジェクトの名前。
type	変更するオブジェクトの型。サポートされている型は、TABLE、INDEX、SYNONYM、TRIGGER、VIEW、PROCEDURE、FUNCTION、PACKAGE および PACKAGE BODY です。
ddl_text	オブジェクトの変更に使用する DDL テキスト。この DDL は、適用前には解析されません。したがって、変更するオブジェクトに対応する適切なスキーマ名とオブジェクト名を DDL テキストで使用していることを確認してください。
comment	NULL 以外の場合に、RepObject ビューの COMMENT フィールドに追加されるコメント。
retry	retry が TRUE の場合は、オブジェクトの状態が VALID 以外のマスターにおいてのみ、オブジェクトが ALTER_MASTER_REPOBJECT によって変更されます。

例外

表 34-15 ALTER\_MASTER\_REPOBJECT プロシージャの例外

例外	説明
nonmasterdef	起動サイトがマスター定義サイトではありません。
notquiesced	関連するオブジェクト・グループが中断されていません。
missingobject	SNAME と ONAME に該当するオブジェクトが存在しません。
typefailure	指定した型パラメータはサポートされていません。
ddlfailure	マスター定義サイトで DDL が成功しませんでした。
commfailure	アクセスできないマスター・サイトが少なくとも 1 つあります。

使用上の注意

スキーマを指定せずに DDL を発行すると、レプリケーション管理者のスキーマがデフォルトのスキーマとして使用されます。レプリケーション管理者のスキーマを使用しない場合は、スキーマを必ず指定してください。

## ALTER\_PRIORITY プロシージャ

このプロシージャは、指定した優先順位グループ・メンバーに関連付けられている優先順位レベルを変更します。このプロシージャは、マスター定義サイトからコールする必要があります。

**関連項目：**『Oracle8i レプリケーション・ガイド』の「競合の解消」を参照してください。

### 構文

```
DBMS_REPCAT.ALTER_PRIORITY (  
    gname          IN   VARCHAR2,  
    pgroup         IN   VARCHAR2,  
    old_priority   IN   NUMBER,  
    new_priority   IN   NUMBER);
```

### パラメータ

表 34-16 ALTER\_PRIORITY プロシージャのパラメータ

パラメータ	説明
gname	優先順位グループが関連付けられているレプリケート・オブジェクト・グループ。
pgroup	変更する優先順位を含んだ優先順位グループの名前。
old_priority	優先順位グループ・メンバーの現在の優先順位レベル。
new_priority	優先順位グループ・メンバーに割り当てる新しい優先順位レベル。

### 例外

表 34-17 ALTER\_PRIORITY プロシージャの例外

例外	説明
nonmasterdef	起動サイトがマスター定義サイトではありません。
duplicatepriority	新しい優先順位レベルが、優先順位グループ内にすでに存在しています。
missingrepgroup	指定したレプリケート・オブジェクト・グループが存在しません。
missingvalue	値が、DBMS_REPCAT.ADD_PRIORITY_datatype のコールで登録されていません。
missingprioritygroup	指定した優先順位グループが存在しません。

表 34-17 ALTER\_PRIORITY プロシージャの例外

例外	説明
notquiesced	指定したレプリケート・オブジェクト・グループが休止状態ではありません。

ALTER\_PRIORITY\_datatype プロシージャ

このプロシージャは、優先順位グループ内のメンバーの値を変更します。このプロシージャは、マスター定義サイトからコールする必要があります。コールする必要があるプロシージャは、priority 列のデータ型によって決まります。

**関連項目：** 詳細は、『Oracle8i レプリケーション・ガイド』の「競合の解消」を参照してください。

構文

```
DBMS_REPCAT.ALTER_PRIORITY_datatype (  
  gname          IN   VARCHAR2,  
  pgroup         IN   VARCHAR2,  
  old_value      IN   datatype,  
  new_value      IN   datatype);
```

datatype には次のいずれかを指定します。

```
{ NUMBER  
| VARCHAR2  
| CHAR  
| DATE  
| RAW  
| NCHAR  
| NVARCHAR2 }
```

パラメータ

表 34-18 ALTER\_PRIORITY\_datatype プロシージャのパラメータ

パラメータ	説明
gname	優先順位グループが関連付けられているレプリケート・オブジェクト・グループ。
pgroup	変更する値を含んだ優先順位グループの名前。
old_value	優先順位グループ・メンバーの現在の値。
new_value	優先順位グループ・メンバーに割り当てる新規の値。



例外

表 34-19 ALTER\_PRIORITY\_datatype プロシージャの例外

例外	説明
nonmasterdef	起動サイトがマスター定義サイトではありません。
duplicatevalue	新しい値が、優先順位グループ内にすでに存在しています。
missingrepgroup	指定したレプリケート・オブジェクト・グループが存在しません。
missingprioritygroup	指定した優先順位グループが存在しません。
missingvalue	元の値が存在しません。
paramtype	優先順位グループに対する新しい値のデータ型が正しくありません。
typefailure	優先順位グループに対して指定した値のデータ型が正しくありません。
notquiesced	指定したレプリケート・オブジェクト・グループが休止状態ではありません。

ALTER\_SITE\_PRIORITY プロシージャ

このプロシージャは、指定したサイトに関連付けられている優先順位レベルを変更します。  
このプロシージャは、マスター定義サイトからコールする必要があります。

**関連項目：**『Oracle8i レプリケーション・ガイド』の「競合の解消」を参照してください。

構文

```
DBMS_REPCAT.ALTER_SITE_PRIORITY (  
  gname          IN   VARCHAR2,  
  name           IN   VARCHAR2,  
  old_priority    IN   NUMBER,  
  new_priority    IN   NUMBER);
```

パラメータ

表 34-20 ALTER\_SITE\_PRIORITY プロシージャのパラメータ

パラメータ	説明
gname	サイト優先順位グループが関連付けられているレプリケート・オブジェクト・グループ。
name	メンバーを変更するサイト優先順位グループの名前。
old_priority	優先順位レベルを変更するサイトの現在の優先順位レベル。
new_priority	サイトの新しい優先順位レベル。数値が大きいほど優先順位レベルは高くなります。

例外

表 34-21 ALTER\_SITE\_PRIORITY プロシージャの例外

例外	説明
nomasterdef	起動サイトがマスター定義サイトではありません。
missingrepgroup	指定したレプリケート・オブジェクト・グループが存在しません。
missingpriority	元の優先順位レベルが、どのグループ・メンバーにも関連付けられていません。
duplicatepriority	新しい優先順位レベルが、グループ内の別のサイト用にすでに存在しています。
missingvalue	元の値が存在しません。
paramtype	優先順位グループに対する新しい値のデータ型が正しくありません。
notquiesced	レプリケート・オブジェクト・グループが休止状態ではありません。

ALTER\_SITE\_PRIORITY\_SITE プロシージャ

このプロシージャは、指定した優先順位レベルに関連付けられているサイトを変更します。  
このプロシージャは、マスター定義サイトからコールする必要があります。

**関連項目：**『Oracle8i レプリケーション・ガイド』の「競合の解消」を参照してください。

## 構文

```
DBMS_REPCAT.ALTER_SITE_PRIORITY_SITE (
    gname      IN   VARCHAR2,
    name       IN   VARCHAR2,
    old_site   IN   VARCHAR2,
    new_site   IN   VARCHAR2);
```

## パラメータ

**表 34-22 ALTER\_SITE\_PRIORITY\_SITE プロシージャのパラメータ**

パラメータ	説明
gname	サイト優先順位グループが関連付けられているレプリケート・オブジェクト・グループ。
name	メンバーを変更するサイト優先順位グループの名前。
old_site	優先順位レベルから分離するサイトの現在のグローバル・データベース名。
new_site	現行優先順位レベルと関連付ける新しいグローバル・データベース名。

## 例外

**表 34-23 ALTER\_SITE\_PRIORITY\_SITE プロシージャの例外**

例外	説明
nonmasterdef	起動サイトがマスター定義サイトではありません。
missingrepgroup	指定したレプリケート・オブジェクト・グループが存在しません。
missingpriority	指定したサイト優先順位グループが存在しません。
missingvalue	旧サイトがグループ・メンバーではありません。
notquiesced	レプリケート・オブジェクト・グループが休止状態ではありません。

## ALTER\_SNAPSHOT\_PROPAGATION プロシージャ

このプロシージャは、現行スナップショット・サイトで指定したオブジェクト・グループの伝播方法を変更します。また、スナップショット・サイトで遅延トランザクション・キューを送信し、スナップショットの実表をロックして、トリガーと関連するパッケージを再生成します。このプロシージャは、スナップショット・サイトからコールする必要があります。

構文

```
DBMS_REPCAT.ALTER_SNAPSHOT_PROPAGATION (  
    gname                IN  VARCHAR2,  
    propagation_mode     IN  VARCHAR2,  
    comment               IN  VARCHAR2    := '');
```

パラメータ

表 34-24 ALTER\_SNAPSHOT\_PROPAGATION プロシージャのパラメータ

パラメータ	説明
gname	伝播モードを変更するオブジェクト・グループの名前。
propagation_mode	現行スナップショット・サイトの変更内容を、関連するマスター・サイトに伝播する方法。指定できる値は SYNCHRONOUS と ASYNCHRONOUS です。
comment	RepProp ビューに追加されるコメント。

例外

表 34-25 ALTER\_SNAPSHOT\_PROPAGATION プロシージャの例外

例外	説明
missingrepgroup	指定したレプリケート・オブジェクト・グループが存在しません。
typefailure	伝播モードの指定に誤りがあります。
nonsnapshot	現行サイトが、指定したオブジェクト・グループのスナップショット・サイトではありません。
commfailure	マスターに接続できません。

CANCEL\_STATISTICS プロシージャ

このプロシージャは、表の更新競合、一意性競合および削除競合の正常な解消に関する統計の収集を停止します。

構文

```
DBMS_REPCAT.CANCEL_STATISTICS (  
    sname    IN  VARCHAR2,  
    oname    IN  VARCHAR2);
```

## パラメータ

表 34-26 CANCEL\_STATISTICS プロシージャのパラメータ

パラメータ	説明
sname	表が置かれているスキーマの名前。
oname	競合解消統計の収集を停止する表の名前。

## 例外

表 34-27 CANCEL\_STATISTICS プロシージャの例外

例外	説明
missingschema	指定したスキーマが存在しません。
missingobject	指定した表が存在しません。
statnotreg	指定した表は、現在統計収集をするために登録されていません。

## COMMENT\_ON\_COLUMN\_GROUP プロシージャ

このプロシージャは、列グループの RepColumn\_Group ビューのコメント・フィールドを更新します。このコメントは、次回の DBMS\_REPCAT.GENERATE\_REPLICATION\_SUPPORT コール後に、すべてのマスター・サイトで追加されます。

## 構文

```
DBMS_REPCAT.COMMENT_ON_COLUMN_GROUP (  
    sname          IN   VARCHAR2,  
    oname          IN   VARCHAR2,  
    column_group   IN   VARCHAR2,  
    comment        IN   VARCHAR2);
```

パラメータ

表 34-28 COMMENT\_ON\_COLUMN\_GROUP プロシージャのパラメータ

パラメータ	説明
sname	オブジェクトが置かれているスキーマの名前。
oname	列グループが関連付けられているレプリケート表の名前。
column_group	列グループの名前。
comment	RepColumn_Group ビューの GROUP_COMMENT フィールドに入れる、更新済みのコメントのテキスト。

例外

表 34-29 COMMENT\_ON\_COLUMN\_GROUP プロシージャの例外

例外	説明
nonmasterdef	起動サイトがマスター定義サイトではありません。
missinggroup	指定した列グループが存在しません。
missingobj	オブジェクトが見つかりません。

COMMENT\_ON\_PRIORITY\_GROUP/PRIORITY プロシージャ

COMMENT\_ON\_PRIORITY\_GROUP は、優先順位グループの REPPRIORITY\_GROUP ビューのコメント・フィールドを更新します。このコメントは、次回の GENERATE\_REPLICATION\_SUPPORT コール後に、すべてのマスター・サイトに追加されます。

COMMENT\_ON\_SITE\_PRIORITY は、サイト優先順位グループの REPPRIORITY\_GROUP ビューのコメント・フィールドを更新します。このプロシージャは、COMMENT\_ON\_COLUMN\_GROUP プロシージャに対するラッパーで、便宜上の目的で提供されています。このプロシージャは、マスター定義サイトで発行する必要があります。

構文

```
DBMS_REPCAT.COMMENT_ON_PRIORITY_GROUP (  
  gname      IN  VARCHAR2,  
  pgroup     IN  VARCHAR2,  
  comment    IN  VARCHAR2);  
  
DBMS_REPCAT.COMMENT_ON_SITE_PRIORITY (  
  gname      IN  VARCHAR2,  
  name       IN  VARCHAR2,  
  comment    IN  VARCHAR2);
```

## パラメータ

**表 34-30 COMMENT\_ON\_PRIORITY\_GROUP/COMMENT\_ON\_SITE\_PRIORITY プロシージャのパラメータ**

パラメータ	説明
gname	レプリケート・オブジェクト・グループの名前。
pgroup/name	優先順位グループまたはサイト優先順位グループの名前。
comment	RepPriority_Group ビューの PRIORITY_COMMENT フィールドに入れる、更新済みのコメントのテキスト。

## 例外

**表 34-31 COMMENT\_ON\_PRIORITY\_GROUP/COMMENT\_ON\_SITE\_PRIORITY プロシージャの例外**

例外	説明
nonmasterdef	起動サイトがマスター定義サイトではありません。
missingrepgroup	指定したレプリケート・オブジェクト・グループが存在しません。
missingprioritygroup	指定した優先順位グループが存在しません。

## COMMENT\_ON\_REPGROUP プロシージャ

このプロシージャは、レプリケート・オブジェクト・グループの REPGROUP ビューのコメント・フィールドを更新します。このプロシージャは、マスター定義サイトで発行する必要があります。

## 構文

```
DBMS_REPCAT.COMMENT_ON_REPGROUP (
    gname      IN   VARCHAR2,
    comment    IN   VARCHAR2);
```

## パラメータ

**表 34-32 COMMENT\_ON\_REPGROUP プロシージャのパラメータ**

パラメータ	説明
gname	コメントを記述するオブジェクト・グループの名前。
comment	RepGroup ビューの SCHEMA_COMMENT フィールドに入れる、更新済みのコメント。

例外

表 34-33 COMMENT\_ON\_REPGROUP プロシージャの例外

例外	説明
nonmasterdef	起動サイトがマスター定義サイトではありません。
commfailure	アクセスできないマスター・サイトが少なくとも 1 つあります。

COMMENT\_ON\_REPSITES プロシージャ

このプロシージャは、レプリケート・サイトの RepSite ビューのコメント・フィールドを更新します。このプロシージャは、マスター定義サイトで発行する必要があります。

構文

```
DBMS_REPCAT.COMMENT_ON_REPSITES (  
  gname      IN   VARCHAR2,  
  [ master   IN   VARCHAR, ]  
  comment    IN   VARCHAR2);
```

パラメータ

表 34-34 COMMENT\_ON\_REPSITES プロシージャのパラメータ

パラメータ	説明
gname	オブジェクト・グループの名前。データベースが複数のレプリケート環境のマスター・サイトである場合は、このパラメータによって混乱を避けることができます。
master	(オプション) コメントを記述するマスター・サイトの完全修飾データベース名。スナップショット・サイトのコメントを更新するときは、このパラメータを省略します。
comment	RepSites ビューの MASTER_COMMENT フィールドに入れる、更新済みのコメントのテキスト。

例外

表 34-35 COMMENT\_ON\_REPSITES プロシージャの例外

例外	説明
nonmasterdef	起動サイトがマスター定義サイトではありません。
nonmaster	起動サイトがマスター・サイトではありません。



表 34-35 COMMENT\_ON\_REPSITES プロシージャの例外

例外	説明
commfailure	アクセスできないマスター・サイトが少なくとも 1 つあります。

## COMMENT\_ON\_REPOBJECT プロシージャ

このプロシージャは、レプリケート・オブジェクトの RepObject ビューのコメント・フィールドを更新します。このプロシージャは、マスター定義サイトで発行する必要があります。

### 構文

```
DBMS_REPCAT.COMMENT_ON_REPOBJECT (  
    sname      IN   VARCHAR2,  
    oname      IN   VARCHAR2,  
    type       IN   VARCHAR2,  
    comment    IN   VARCHAR2);
```

### パラメータ

表 34-36 COMMENT\_ON\_REPOBJECT プロシージャのパラメータ

パラメータ	説明
sname	オブジェクトが置かれているスキーマの名前。
oname	コメントを記述するオブジェクトの名前。
type	オブジェクトの型。
comment	RepObject ビューの OBJECT_COMMENT フィールドに入れる、更新済みのコメントのテキスト。

### 例外

表 34-37 COMMENT\_ON\_REPOBJECT プロシージャの例外

例外	説明
nonmasterdef	起動サイトがマスター定義サイトではありません。
missingobject	指定したオブジェクトが存在しません。
typefailure	指定した型パラメータはサポートされていません。
commfailure	アクセスできないマスター・サイトが少なくとも 1 つあります。

## COMMENT\_ON\_conflicttype\_RESOLUTION プロシージャ

このプロシージャは、競合解消ルーチンの RepResolution ビューのコメント・フィールドを更新します。コールする必要があるプロシージャは、ルーチンが解消する競合の型によって決まります。このプロシージャは、マスター定義サイトで発行する必要があります。

競合の型	プロシージャ名
更新	COMMENT_ON_UPDATE_RESOLUTION
一意性	COMMENT_ON_UNIQUE_RESOLUTION
削除	COMMENT_ON_DELETE_RESOLUTION

コメントは、次回の GENERATE\_REPLICATION\_SUPPORT コール後に、すべてのマスター・サイトに追加されます。

### 構文

```
DBMS_REPCAT.COMMENT_ON_UPDATE_RESOLUTION (
    sname          IN   VARCHAR2,
    oname          IN   VARCHAR2,
    column_group   IN   VARCHAR2,
    sequence_no    IN   NUMBER,
    comment        IN   VARCHAR2);

DBMS_REPCAT.COMMENT_ON_UNIQUE_RESOLUTION (
    sname          IN   VARCHAR2,
    oname          IN   VARCHAR2,
    constraint_name IN   VARCHAR2,
    sequence_no    IN   NUMBER,
    comment        IN   VARCHAR2);

DBMS_REPCAT.COMMENT_ON_DELETE_RESOLUTION (
    sname          IN   VARCHAR2,
    oname          IN   VARCHAR2,
    sequence_no    IN   NUMBER,
    comment        IN   VARCHAR2);
```

## パラメータ

**表 34-38 COMMENT\_ON\_conflictype\_RESOLUTION プロシージャのパラメータ**

パラメータ	説明
sname	スキーマ名。
oname	競合解消ルーチンが関連付けられているレプリケート表の名前。
column_group	更新競合解消ルーチンが関連付けられている列グループの名前。
constraint_name	一意性競合解消ルーチンが関連付けられている一意制約の名前。
sequence_no	競合解消プロシージャの順序番号。
comment	RepResolution ビューの RESOLUTION_COMMENT フィールドに入る、更新済みのコメントのテキスト。

## 例外

**表 34-39 COMMENT\_ON\_conflictype\_RESOLUTION プロシージャの例外**

例外	説明
nonmasterdef	起動サイトがマスター定義サイトではありません。
missingobject	指定したオブジェクトが存在しません。
missingresolution	指定した競合解消ルーチンが登録されていません。

## COMPARE\_OLD\_VALUES プロシージャ

更新と削除のために、レプリケート表の各非キー列の元の値を比較するオプションがあります。デフォルトでは、すべての列の元の値が比較されます。DBMS\_REPCAT.COMPARE\_OLD\_VALUES をマスター定義サイトで起動すると、すべてのマスター・サイトとスナップショット・サイトでこの動作を変更できます。

## 構文

```
DBMS_REPCAT.COMPARE_OLD_VALUES(
    sname          IN  VARCHAR2,
    oname          IN  VARCHAR2,
    { column_list  IN  VARCHAR2
    | column_table IN  DBMS_REPCAT.VARCHAR2s, }
    operation      IN  VARCHAR2 := 'UPDATE',
    compare        IN  BOOLEAN := TRUE );
```

**注意：** このプロシージャはオーバーロードされています。column\_list パラメータと column\_table パラメータは、両方同時には指定できません。

パラメータ

表 34-40 COMPARE\_OLD\_VALUES プロシージャのパラメータ

パラメータ	説明
sname	表が置かれているスキーマ。
oname	レプリケート表の名前。
column_list	表内の列のカンマで区切られたリスト。エントリ間に空白を挿入しないでください。
column_table	リストのかわりに、列名を含んだ型 DBMS_REPCAT.VARCHAR2 の PL/SQL 表を使用できます。最初の列名はオフセット 1 の位置、2 番目はオフセット 2 の位置、以下同様に設定されている必要があります。
operation	有効な値は、UPDATE、DELETE、またはアスタリスクのワイルドカード '*'（更新と削除を意味します）です。
compare	TRUE の場合は、指定した列の元の値が送信時に比較されます。FALSE の場合は、指定した列の元の値が送信時に比較されません。指定外の列と指定外の操作には影響しません。マスター定義サイトでは、表の min_communication が TRUE になると、指定した変更がすぐに有効となります。変更内容がマスター・サイトまたはスナップショット・サイトで有効になるのは、min_communication に TRUE を設定して、次回そのサイトでレプリケーション・サポートを生成したときです。

**注意：** operation パラメータを使用すると、行の削除時または非キー列の更新時に、非キー列の元の値を送信するかどうかを決定できます。元の値を送信しないと、元の値のかわりに NULL が送信され、更新または削除の適用時にターゲット側の現在の列値と元の値が等しいとみなされます。

Oracle のデフォルト動作を変更するには、『Oracle8i レプリケーション・ガイド』の「更新の競合解消のためのデータ伝播の最少化」を参照してください。

## 例外

表 34-41 COMPARE\_OLD\_VALUES プロシージャの例外

例外	説明
nonmasterdef	起動サイトがマスター定義サイトではありません。
missingobject	指定したオブジェクトは、行レベル・レプリケーション情報を待機している表として指定のスキーマ内に存在していません。
missingcolumn	表に存在しない列が少なくとも 1 つあります。
notquiesced	レプリケート・オブジェクト・グループが中断されていません。
typefailure	無効な操作が指定されています。

## CREATE\_MASTER\_REPGROUP プロシージャ

このプロシージャは、新規で、空で、休止状態のマスター・レプリケーション・オブジェクト・グループを作成します。

### 構文

```
DBMS_REPCAT.CREATE_MASTER_REPGROUP (
    gname          IN   VARCHAR2,
    group_comment  IN   VARCHAR2    := '',
    master_comment IN   VARCHAR2    := ''),
    qualifier      IN   VARCHAR2    := '');
```

## パラメータ

表 34-42 CREATE\_MASTER\_REPGROUP プロシージャのパラメータ

パラメータ	説明
gname	作成するオブジェクト・グループの名前。
group_comment	RepGroup ビューに追加されるコメント。
master_comment	RepSites ビューに追加されるコメント。
qualifier	オブジェクト・グループの接続修飾子。必ず @ 符号を使用してください。『Oracle8i レプリケーション・ガイド』の「マスター・グループの管理」の例を参照してください。

例外

表 34-43 CREATE\_MASTER\_REPGROUP プロシージャの例外

例外	説明
duplicaterepgroup	オブジェクト・グループはすでに存在しています。
norepopt	アドバンスド・レプリケーション・オプションがインストールされていません。
missingrepgroup	オブジェクト・グループ名が指定されていません。
qualifiertoolong	接続修飾子が長すぎます。

CREATE\_MASTER\_REPOBJECT プロシージャ

このプロシージャは、オブジェクトをレプリケート・オブジェクトとして定義します。

構文

```
DBMS_REPCAT.CREATE_MASTER_REPOBJECT (  
  sname          IN   VARCHAR2,  
  oname          IN   VARCHAR2,  
  type           IN   VARCHAR2,  
  use_existing_object IN  BOOLEAN      := TRUE,  
  ddl_text       IN   VARCHAR2      := NULL,  
  comment        IN   VARCHAR2      := '',  
  retry          IN   BOOLEAN      := FALSE,  
  copy_rows      IN   BOOLEAN      := TRUE,  
  gname          IN   VARCHAR2      := '');
```

## パラメータ

表 34-44 CREATE\_MASTER\_REPOBJECT プロシージャのパラメータ

パラメータ	説明
sname	レプリケートするオブジェクトが置かれているスキーマの名前。
oname	レプリケートするオブジェクトの名前。DDL_TEXT が NULL の場合は、指定したスキーマ内にこのオブジェクトがすでに存在している必要があります。一意性を確保するために、表名は 27 バイト以内、パッケージ名は 24 バイト以内で指定してください。
type	レプリケートするオブジェクトの型。サポートされている型は、TABLE、INDEX、SYNONYM、TRIGGER、VIEW、PROCEDURE、FUNCTION、PACKAGE および PACKAGE BODY です。
use_existing_object	現行マスター・サイトの同じ型、同じ形式のオブジェクトを再使用する場合は、TRUE を指定します。詳細は、表 34-46 を参照してください。
ddl_text	オブジェクトがマスター定義サイトにまだ存在していない場合は、このオブジェクトの作成に必要な DDL テキストを指定します。PL/SQL パッケージ、パッケージ本体、プロシージャおよびファンクションの後には、セミコロンを付ける必要があります。SQL 文には、セミコロンを付ける必要はありません。この DDL は、適用前には解析されません。したがって、作成するオブジェクトに対応する適切なスキーマ名とオブジェクト名を DDL テキストで使用していることを確認してください。
comment	RepObject ビューの OBJECT_COMMENT フィールドに追加されるコメント。
retry	以前にオブジェクトを作成できなかった場合に、そのオブジェクトの作成を再試行する場合は、TRUE を指定します。このパラメータは、エラーが一時的な場合やすでに修正されている場合（リソースが不足などのエラーの場合）に使用します。TRUE の場合、オブジェクトは、オブジェクトの状態が VALID 以外のマスター・サイトでのみ作成されます。
copy_rows	新規レプリケート・オブジェクトの初期の内容を、マスター定義サイトのオブジェクトの内容と一致させる場合は TRUE を指定します。詳細は、表 34-46 を参照してください。
gname	レプリケート・オブジェクトを作成するオブジェクト・グループの名前。指定しないと、デフォルトのオブジェクト・グループ名としてスキーマ名が使用されます。

表 34-45 CREATE\_MASTER\_REOBJECT プロシージャの例外

例外	説明
nomasterdef	起動サイトがマスター定義サイトではありません。
notquiesced	レプリケート・オブジェクト・グループが中断されていません。
duplicateobject	指定したオブジェクトがレプリケート・オブジェクト・グループにすでに存在しており、retry が FALSE であるか、または名前の競合が発生しています。
missingobject	SNAME と ONAME に該当するオブジェクトが存在せず、適切な DDL も指定されていません。
typefailure	指定した型のオブジェクトはレプリケートできません。
ddlfailure	マスター定義サイトで DDL が成功しませんでした。
commfailure	アクセスできないマスター・サイトが少なくとも 1 つあります。
notcompat	7.3 互換モードではないリモート・マスターがあります。

オブジェクト作成

表 34-46 マスター・サイトにおけるオブジェクト作成

オブジェクトはすでに存在	COPY_ROWS	USE_EXISTING_OBJECTS	結果
はい	TRUE	TRUE	オブジェクトが一致しない場合は duplicatedobject メッセージが戻されます。表の場合は、マスター定義サイトのデータが使用されます。
はい	FALSE	TRUE	オブジェクトが一致しない場合は duplicatedobject メッセージが戻されます。表の場合、DBA は内容が同じであることを確認する必要があります。
はい	TRUE/FALSE	FALSE	duplicatedobject メッセージが戻されます。
いいえ	TRUE	TRUE/FALSE	オブジェクトが作成されます。表には、マスター定義サイトのデータが移入されます。
いいえ	FALSE	TRUE/FALSE	オブジェクトが作成されます。DBA は表にデータを移入し、すべてのサイトで表の内容が一致していることを確認する必要があります。



使用上の注意

スキーマを指定せずに DDL を発行すると、レプリケーション管理者のスキーマがデフォルトのスキーマとして使用されます。レプリケーション管理者のスキーマを使用しない場合は、スキーマを必ず指定してください。

CREATE\_SNAPSHOT\_REPGROUP プロシージャ

このプロシージャは、新規で空のスナップショット・レプリケーション・オブジェクト・グループをローカル・データベース内に作成します。

構文

```
DBMS_REPCAT.CREATE_SNAPSHOT_REPGROUP (  
    gname          IN   VARCHAR2,  
    master         IN   VARCHAR2,  
    comment        IN   VARCHAR2      := '',  
    propagation_mode IN VARCHAR2      := 'ASYNCHRONOUS',  
    fname         IN   VARCHAR2      := NULL);
```

パラメータ

表 34-47 CREATE\_SNAPSHOT\_REPGROUP プロシージャのパラメータ

パラメータ	説明
gname	レプリケート・オブジェクト・グループの名前。このオブジェクト・グループは、指定したマスター・サイトに存在している必要があります。
master	レプリケート環境でマスターとして使用するデータベースの完全修飾データベース名。
comment	RepGroup ビューに追加されるコメント。
propagation_mode	オブジェクト・グループ内のすべての更新可能スナップショットに使用する伝播方法。指定できる値は SYNCHRONOUS と ASYNCHRONOUS です。
fname	内部使用のためのシステム・パラメータ。オラクル社カスタマ・サポートから指示がない限り、このパラメータは設定しないでください。

例外

表 34-48 CREATE\_SNAPSHOT\_REPGROUP プロシージャの例外

例外	説明
duplicaterepgroup	オブジェクト・グループが、起動サイトにすでに存在しています。
nonmaster	指定したデータベースはマスター・サイトではありません。
commfailure	指定したデータベースにアクセスできません。
norepopt	アドバンスト・レプリケーション・オプションがインストールされていません。
typefailure	伝播モードの指定に誤りがあります。
missingrepgroup	レプリケート・オブジェクト・グループがマスター・サイトに存在していません。

使用上の注意

CREATE\_SNAPSHOT\_REPGROUP は、REGISTER\_SNAPSHOT\_REPGROUP を自動的にコールします。ただし、登録中にエラーが発生しても無視されます。

CREATE\_SNAPSHOT\_REPOBJECT プロシージャ

このプロシージャは、スナップショット・サイトにレプリケート・オブジェクトを追加します。

構文

```
DBMS_REPCAT.CREATE_SNAPSHOT_REPOBJECT (
    sname          IN   VARCHAR2,
    oname          IN   VARCHAR2,
    type           IN   VARCHAR2,
    ddl_text       IN   VARCHAR2  := '',
    comment        IN   VARCHAR2  := '',
    gname          IN   VARCHAR2  := '',
    gen_objs_owner IN   VARCHAR2  := '',
    min_communication IN BOOLEAN  := TRUE ,
    generate_80_compatible IN BOOLEAN  := TRUE);
```

## パラメータ

**表 34-49 CREATE\_SNAPSHOT\_REPOBJECT プロシージャのパラメータ**

パラメータ	説明
sname	オブジェクトが置かれているスキーマの名前。
oname	レプリケート・スナップショット・オブジェクト・グループに追加するオブジェクトの名前。ONAME が関連するマスター・サイトに存在している必要があります。
type	レプリケートするオブジェクトの型。スナップショット・サイトでサポートされる型は、PACKAGE、PACKAGE BODY、PROCEDURE、FUNCTION、SNAPSHOT、SYNONYM、TRIGGER および VIEW です。
ddl_text	SNAPSHOT 型のオブジェクトの場合は、オブジェクトの作成に必要な DDL テキストを指定します。その他の型の場合は、デフォルトの '' (空文字列) を使用します。同じ名前のスナップショットがすでに存在している場合、DDL は無視され、既存のスナップショットがレプリケート・オブジェクトとして登録されます。スナップショットのマスター表が、このスキーマに対して指定したマスター・サイトのレプリケート・オブジェクト・グループ内に存在しない場合は、missingobject エラーが発生します。
comment	RepObject ビューの OBJECT_COMMENT フィールドに追加されるコメント。
gname	オブジェクトを追加するレプリケート・オブジェクト・グループの名前。指定しないと、デフォルトのグループ名としてスキーマ名が使用されます。
gen_objs_owner	トランザクションの所有者として割り当てるユーザーの名前。
min_communication	マスター・サイトのいずれかが Oracle7 リリース 7.3 を実行している場合は、FALSE を設定します。新旧の値の伝播を最小化する場合は、TRUE を設定します。デフォルトは TRUE です。詳細は、『Oracle8i レプリケーション・ガイド』の「競合の解消」を参照してください。
generate_80_compatible	マスター・サイトのいずれかが Oracle8i リリース 8.1.5 以前のバージョンの Oracle Server を実行している場合は、TRUE を設定します。レプリケート環境が純粋な Oracle8i リリース 8.1.5 以上の環境の場合は、FALSE を設定します。

例外

表 34-50 CREATE\_SNAPSHOT\_REPOBJECT プロシージャの例外

例外	説明
nonsnapshot	起動サイトがスナップショット・サイトではありません。
nonmaster	このマスターは、マスター・サイトではありません。
missingobject	指定したオブジェクトがマスターのレプリケート・オブジェクト・グループに存在していません。
duplicateobject	指定したオブジェクトは、すでに別の形式で存在しています。
typefailure	この型は許可されていません。
ddlfailure	DDL は成功しませんでした。
commfailure	マスター・サイトにアクセスできません。
missingschema	スキーマが、データベース・スキーマとして存在していません。
badsnapddl	DDL は実行されましたが、スナップショットが存在していません。
onlyonesnap	マスター表のスナップショットは、1 つしか作成できません。
badsnapname	スナップショットの実表がマスター表と異なります。
missingrepgroup	レプリケート・オブジェクト・グループが存在しません。

使用上の注意

スキーマを指定せずに DDL を発行すると、レプリケーション管理者のスキーマがデフォルトのスキーマとして使用されます。レプリケーション管理者のスキーマを使用しない場合は、スキーマを必ず指定してください。

DEFINE\_COLUMN\_GROUP プロシージャ

このプロシージャは、空の列グループを作成します。このプロシージャは、マスター定義サイトからコールする必要があります。

関連項目： 詳細は、『Oracle8i レプリケーション・ガイド』の「競合の解消」を参照してください。

構文

```
DBMS_REPCAT.DEFINE_COLUMN_GROUP (  
    sname           IN   VARCHAR2,  
    oname           IN   VARCHAR2,  
    column_group    IN   VARCHAR2,  
    comment         IN   VARCHAR2 := NULL);
```

パラメータ

表 34-51 DEFINE\_COLUMN\_GROUP プロシージャのパラメータ

パラメータ	説明
sname	レプリケート表が置かれているスキーマ。
oname	列グループを作成するレプリケート表の名前。
column_group	作成する列グループの名前。
comment	RepColumn_Group ビューに表示されるユーザー・テキスト。

例外

表 34-52 DEFINE\_COLUMN\_GROUP プロシージャの例外

例外	説明
nonmasterdef	起動サイトがマスター定義サイトではありません。
missingobject	指定した表が存在しません。
duplicategroup	指定した列グループは、すでに表に存在しています。
notquiesced	指定した表が属するオブジェクト・グループが休止状態ではありません。

DEFINE\_PRIORITY\_GROUP プロシージャ

このプロシージャは、レプリケート・オブジェクト・グループに新規の優先順位グループを作成します。このプロシージャは、マスター定義サイトからコールする必要があります。

**関連項目：**『Oracle8i レプリケーション・ガイド』の「競合の解消」を参照してください。

構文

```
DBMS_REPCAT.DEFINE_PRIORITY_GROUP (  
    gname          IN   VARCHAR2,  
    pgroup         IN   VARCHAR2,  
    datatype       IN   VARCHAR2,  
    fixed_length   IN   INTEGER := NULL,  
    comment        IN   VARCHAR2 := NULL);
```

パラメータ

表 34-53 DEFINE\_PRIORITY\_GROUP プロシージャのパラメータ

パラメータ	説明
gname	優先順位グループを作成するレプリケート・オブジェクト・グループ。
pgroup	作成する優先順位グループの名前。
datatype	優先順位グループ・メンバーのデータ型。サポートされているデータ型は、CHAR、VARCHAR2、NUMBER、DATE、RAW、NCHAR および NVARCHAR2 です。
fixed_length	CHAR データ型の場合は、列の長さを指定してください。その他のすべての型では、デフォルトの NULL を使用できます。
comment	RepPriority ビューに追加されるユーザー・コメント。

例外

表 34-54 DEFINE\_PRIORITY\_GROUP プロシージャの例外

例外	説明
nonmasterdef	起動サイトがマスター定義サイトではありません。
missingrepgroup	指定したレプリケート・オブジェクト・グループが存在しません。
duplicateprioritygroup	指定した優先順位グループは、レプリケート・オブジェクト・グループ内にすでに存在しています。
typefailure	指定したデータ型はサポートされていません。
notquiesced	レプリケート・オブジェクト・グループが休止状態ではありません。

## DEFINE\_SITE\_PRIORITY プロシージャ

このプロシージャは、レプリケート・オブジェクト・グループに新規のサイト優先順位グループを作成します。このプロシージャは、マスター定義サイトからコールする必要があります。

**関連項目：**『Oracle8i レプリケーション・ガイド』の「競合の解消」を参照してください。

### 構文

```
DBMS_REPCAT.DEFINE_SITE_PRIORITY (
    gname          IN   VARCHAR2,
    name           IN   VARCHAR2,
    comment        IN   VARCHAR2 := NULL);
```

### パラメータ

**表 34-55 DEFINE\_SITE\_PRIORITY プロシージャのパラメータ**

パラメータ	説明
gname	サイト優先順位グループを作成するレプリケート・オブジェクト・グループ。
name	作成するサイト優先順位グループの名前。
comment	RepPriority ビューに追加されるユーザー・コメント。

### 例外

**表 34-56 DEFINE\_SITE\_PRIORITY プロシージャの例外**

例外	説明
nonmasterdef	起動サイトがマスター定義サイトではありません。
missingrepgroup	指定したレプリケート・オブジェクト・グループが存在しません。
duplicate prioritygroup	指定したサイト優先順位グループは、レプリケート・オブジェクト・グループ内にすでに存在しています。
notquiesced	レプリケート・オブジェクト・グループが休止状態ではありません。

## DO\_DEFERRED\_REPCAT\_ADMIN プロシージャ

このプロシージャは、現行マスター・サイトで指定したレプリケート・オブジェクト・グループ、またはすべてのマスター・サイト（ジョブ・キューを利用）に対する未処理のローカル遅延管理プロシージャを実行します。

構文

```
DBMS_REPCAT.DO_DEFERRED_REPCAT_ADMIN (  
    gname          IN   VARCHAR2,  
    all_sites      IN   BOOLEAN := FALSE);
```

パラメータ

表 34-57 DO\_DEFERRED\_REPCAT\_ADMIN プロシージャのパラメータ

パラメータ	説明
gname	レプリケート・オブジェクト・グループの名前。
all_sites	TRUE の場合は、ジョブを使用して各マスターごとにローカル管理プロセスを実行します。

例外

表 34-58 DO\_DEFERRED\_REPCAT\_ADMIN プロシージャの例外

例外	説明
nonmaster	起動サイトがマスター・サイトではありません。
commfailure	all_sites が TRUE ですが、アクセスできないマスター・サイトが少なくとも 1 つあります。

使用上の注意

DO\_DEFERRED\_REPCAT\_ADMIN は、DO\_DEFERRED\_REPCAT\_ADMIN をコールした接続ユーザーが送信した管理要求のみ実行します。その他のユーザーが送信した要求は無視されます。

DROP\_COLUMN\_GROUP プロシージャ

このプロシージャは列グループを削除します。このプロシージャは、マスター定義サイトからコールする必要があります。

**関連項目：**『Oracle8i レプリケーション・ガイド』の「競合の解消」を参照してください。

構文

```
DBMS_REPCAT.DROP_COLUMN_GROUP (  
    sname          IN   VARCHAR2,  
    oname          IN   VARCHAR2,  
    column_group IN   VARCHAR2);
```



## パラメータ

表 34-59 DROP\_COLUMN\_GROUP プロシージャのパラメータ

パラメータ	説明
sname	レプリケート表が置かれているスキーマ。
oname	列グループを削除するレプリケート表の名前。
column_group	削除する列グループの名前。

## 例外

表 34-60 DROP\_COLUMN\_GROUP プロシージャの例外

例外	説明
nonmasterdef	起動サイトがマスター定義サイトではありません。
referenced	指定した列グループは、競合の検出と解消で使用されています。
missingobject	指定した表が存在しません。
missinggroup	指定した列グループが存在しません。
notquiesced	表が属するレプリケート・オブジェクト・グループが休止状態ではありません。

## DROP\_GROUPED\_COLUMN プロシージャ

このプロシージャは、列グループからメンバーを削除します。このプロシージャは、マスター定義サイトからコールする必要があります。

**関連項目：** 詳細は、『Oracle8i レプリケーション・ガイド』の「競合の解消」を参照してください。

## 構文

```
DBMS_REPCAT.DROP_GROUPED_COLUMN (  
  sname          IN   VARCHAR2,  
  oname          IN   VARCHAR2,  
  column_group   IN   VARCHAR2,  
  list_of_column_names IN VARCHAR2 | DBMS_REPCAT.VARCHAR2S);
```

## パラメータ

表 34-61 DROP\_GROUPED\_COLUMN プロシージャのパラメータ

パラメータ	説明
sname	レプリケート表が置かれているスキーマ。
oname	列グループが置かれているレプリケート表の名前。
column_group	メンバーを削除する列グループの名前。
list_of_column_names	指定した列グループから削除する列の名前。列名は、名前のカンマで区切られたリストまたは PL/SQL 表のいずれかで示します。PL/SQL 表は、dbms_repcat.varchar2 型にしてください。

## 例外

表 34-62 DROP\_GROUPED\_COLUMN プロシージャの例外

例外	説明
nonmasterdef	起動サイトがマスター定義サイトではありません。
missingobject	指定した表が存在しません。
notquiesced	表が属するレプリケート・オブジェクト・グループが休止状態ではありません。

## DROP\_MASTER\_REPGROUP プロシージャ

このプロシージャは、現行サイトからレプリケート・オブジェクト・グループを削除します。マスター定義サイトも含め、すべてのマスター・サイトからレプリケート・オブジェクト・グループを削除するには、最後の引数に TRUE を設定して、このプロシージャをマスター定義サイトでコールします。

## 構文

```
DBMS_REPCAT.DROP_MASTER_REPGROUP (  
    gname          IN VARCHAR2,  
    drop_contents  IN BOOLEAN    := FALSE,  
    all_sites      IN BOOLEAN    := FALSE);
```

## パラメータ

表 34-63 DROP\_MASTER\_REPGROUP プロシージャのパラメータ

パラメータ	説明
gname	現行マスター・サイトから削除するレプリケート・オブジェクト・グループの名前。
drop_contents	デフォルトでは、マスター・サイトのオブジェクト・グループを削除したとき、すべてのオブジェクトがデータベースに残ります。このオブジェクトは、この後レプリケートされなくなります。つまり、オブジェクト・グループ内のレプリケート・オブジェクトと他のマスター・サイトとの間で、変更内容の送受信が行われなくなります。このパラメータに TRUE を設定すると、レプリケート・オブジェクト・グループ内のすべてのレプリケート・オブジェクトが、関連するスキーマから削除されます。
all_sites	このパラメータが TRUE で起動サイトがマスター定義サイトの場合、このプロシージャは、要求をすべてのマスターに同期式でマルチキャストします。この場合、マスター定義サイトでは要求がすぐに実行され、その他のマスター・サイトでは遅れて実行される可能性があります。

## 例外

表 34-64 DROP\_MASTER\_REPGROUP プロシージャの例外

例外	説明
nonmaster	起動サイトがマスター・サイトではありません。
nonmasterdef	ALL_SITES が TRUE ですが、起動サイトがマスター定義サイトではありません。
commfailure	ALL_SITES が TRUE ですが、アクセスできないマスター・サイトが少なくとも 1 つあります。
fullqueue	遅延 RPC キューに、レプリケート・オブジェクト・グループに対するエントリがあります。
masternotremoved	ALL_SITES が TRUE ですが、マスターでマスター定義が認識されていません。

## DROP\_MASTER\_REPOBJECT プロシージャ

このプロシージャは、レプリケート・オブジェクト・グループからレプリケート・オブジェクトを削除します。このプロシージャは、マスター定義サイトからコールする必要があります。

### 構文

```
DBMS_REPCAT.DROP_MASTER_REPOBJECT (  
    sname          IN   VARCHAR2,  
    oname          IN   VARCHAR2,  
    type           IN   VARCHAR2,  
    drop_objects   IN   BOOLEAN      := FALSE);
```

### パラメータ

表 34-65 DROP\_MASTER\_REPOBJECT プロシージャのパラメータ

パラメータ	説明
sname	オブジェクトが置かれているスキーマの名前。
oname	レプリケート・オブジェクト・グループから削除するオブジェクトの名前。
type	削除するオブジェクトの型。
drop_objects	デフォルトでは、オブジェクトはスキーマ内に残りますが、レプリケート・オブジェクト・グループからは削除されます。つまり、このオブジェクトに対する変更は、他のマスター・サイトとスナップショット・サイトにレプリケートされなくなります。レプリケート環境からオブジェクトを完全に削除するには、この引数に TRUE を設定します。

### 例外

表 34-66 DROP\_MASTER\_REPOBJECT プロシージャの例外

例外	説明
nonmasterdef	起動サイトがマスター定義サイトではありません。
missingobject	指定したオブジェクトが存在しません。
typefailure	指定した型パラメータはサポートされていません。
commfailure	アクセスできないマスター・サイトが少なくとも 1 つあります。

## DROP\_PRIORITY プロシージャ

このプロシージャは、優先順位レベルに従って優先順位グループのメンバーを削除します。  
このプロシージャは、マスター定義サイトからコールする必要があります。

**関連項目：**『Oracle8i レプリケーション・ガイド』の「競合の解消」を参照してください。

### 構文

```
DBMS_REPCAT.DROP_PRIORITY(  
    gname          IN   VARCHAR2,  
    pgroup         IN   VARCHAR2,  
    priority_num    IN   NUMBER);
```

### パラメータ

表 34-67 DROP\_PRIORITY プロシージャのパラメータ

パラメータ	説明
gname	優先順位グループが関連付けられているレプリケート・オブジェクト・グループ。
pgroup	削除するメンバーを含んだ優先順位グループの名前。
priority_num	グループから削除する優先順位グループ・メンバーの優先順位レベル。

### 例外

表 34-68 DROP\_PRIORITY プロシージャの例外

例外	説明
nonmasterdef	起動サイトがマスター定義サイトではありません。
missingrepgroup	指定したレプリケート・オブジェクト・グループが存在しません。
missingprioritygroup	指定した優先順位グループが存在しません。
notquiesced	レプリケート・オブジェクト・グループが休止状態ではありません。

## DROP\_PRIORITY\_GROUP プロシージャ

このプロシージャは、指定したレプリケート・オブジェクト・グループの優先順位グループを削除します。このプロシージャは、マスター定義サイトからコールする必要があります。

**関連項目：**『Oracle8i レプリケーション・ガイド』の「競合の解消」を参照してください。

構文

```
DBMS_REPCAT.DROP_PRIORITY_GROUP (  
    gname      IN   VARCHAR2,  
    pgroup     IN   VARCHAR2);
```

パラメータ

表 34-69 DROP\_PRIORITY\_GROUP プロシージャのパラメータ

パラメータ	説明
gname	優先順位グループが関連付けられているレプリケート・オブジェクト・グループ。
pgroup	削除する優先順位グループの名前。

例外

表 34-70 DROP\_PRIORITY\_GROUP プロシージャの例外

例外	説明
nonmasterdef	起動サイトがマスター定義サイトではありません。
missingrepgroup	指定したレプリケート・オブジェクト・グループが存在しません。
referenced	指定した優先順位グループは、競合の解消で使用されています。
notquiesced	指定したレプリケート・オブジェクト・グループが休止状態ではありません。

DROP\_PRIORITY\_datatype プロシージャ

このプロシージャは、値による優先順位グループのメンバーを削除します。このプロシージャは、マスター定義サイトからコールする必要があります。コールする必要があるプロシージャは、priority 列のデータ型によって決まります。

**関連項目：**『Oracle8i レプリケーション・ガイド』の「競合の解消」を参照してください。

構文

```
DBMS_REPCAT.DROP_PRIORITY_datatype (  
    gname      IN   VARCHAR2,  
    pgroup     IN   VARCHAR2,  
    value      IN   datatype);
```

datatype には次のいずれかを指定します。

```
{ NUMBER  
| VARCHAR2  
| CHAR  
| DATE  
| RAW  
| NCHAR  
| NVARCHAR2 }
```

パラメータ

表 34-71 DROP\_PRIORITY\_datatype プロシージャのパラメータ

パラメータ	説明
gname	優先順位グループが関連付けられているレプリケート・オブジェクト・グループ。
pgroup	削除するメンバーを含んだ優先順位グループの名前。
value	グループから削除する優先順位グループ・メンバーの値。

例外

表 34-72 DROP\_PRIORITY\_datatype プロシージャの例外

例外	説明
nonmasterdef	起動サイトがマスター定義サイトではありません。
missingrepgroup	指定したレプリケート・オブジェクト・グループが存在しません。
missingprioritygroup	指定した優先順位グループが存在しません。
paramtype, typefailure	優先順位グループに対する値のデータ型が正しくありません。
notquiesced	指定したレプリケート・オブジェクト・グループが休止状態ではありません。

## DROP\_SITE\_PRIORITY プロシージャ

このプロシージャは、指定したレプリケート・オブジェクト・グループのサイト優先順位グループを削除します。このプロシージャは、マスター定義サイトからコールする必要があります。

**関連項目：**『Oracle8i レプリケーション・ガイド』の「競合の解消」を参照してください。

### 構文

```
DBMS_REPCAT.DROP_SITE_PRIORITY (  
    gname      IN    VARCHAR2,  
    name       IN    VARCHAR2);
```

### パラメータ

表 34-73 DROP\_SITE\_PRIORITY プロシージャのパラメータ

パラメータ	説明
gname	サイト優先順位グループが関連付けられているレプリケート・オブジェクト・グループ。
name	削除するサイト優先順位グループの名前。

### 例外

表 34-74 DROP\_SITE\_PRIORITY プロシージャの例外

例外	説明
nonmasterdef	起動サイトがマスター定義サイトではありません。
missingrepgroup	指定したレプリケート・オブジェクト・グループが存在しません。
referenced	指定したサイト優先順位グループは、競合の解消で使用されています。
notquiesced	指定したレプリケート・オブジェクト・グループが休止状態ではありません。

## DROP\_SITE\_PRIORITY\_SITE プロシージャ

このプロシージャは、サイト優先順位グループから、名前によって指定したサイトを削除します。このプロシージャは、マスター定義サイトからコールする必要があります。



**関連項目：**『Oracle8i レプリケーション・ガイド』の「競合の解消」を参照してください。

## 構文

```
DBMS_REPCAT.DROP_SITE_PRIORITY_SITE (
    gname      IN   VARCHAR2,
    name       IN   VARCHAR2,
    site       IN   VARCHAR2);
```

## パラメータ

**表 34-75 DROP\_SITE\_PRIORITY\_SITE プロシージャのパラメータ**

パラメータ	説明
gname	サイト優先順位グループが関連付けられているレプリケート・オブジェクト・グループ。
name	メンバーを削除するサイト優先順位グループの名前。
site	グループから削除するサイトのグローバル・データベース名。

## 例外

**表 34-76 DROP\_SITE\_PRIORITY\_SITE プロシージャの例外**

例外	説明
nonmasterdef	起動サイトがマスター定義サイトではありません。
missingrepgroup	指定したレプリケート・オブジェクト・グループが存在しません。
missingpriority	指定したサイト優先順位グループが存在しません。
notquiesced	指定したレプリケート・オブジェクト・グループが休止状態ではありません。

## DROP\_SNAPSHOT\_REPGROUP プロシージャ

このプロシージャは、レプリケート環境からスナップショット・サイトを削除します。

## 構文

```
DBMS_REPCAT.DROP_SNAPSHOT_REPGROUP (
    gname      IN   VARCHAR2,
    drop_contents  IN   BOOLEAN    := FALSE);
```

パラメータ

表 34-77 DROP\_SNAPSHOT\_REPGROUP プロシージャのパラメータ

パラメータ	説明
gname	現行スナップショット・サイトから削除するレプリケート・オブジェクト・グループの名前。トリガーやパッケージなど、レプリケーションをサポートするために生成されたすべてのオブジェクトが削除されます。
drop_contents	デフォルトでは、スナップショット・サイトのレプリケート・オブジェクト・グループを削除したとき、オブジェクトはすべて関連するスキーマに残ります。つまり、このオブジェクトはレプリケートされなくなります。このパラメータに TRUE を設定すると、レプリケート・オブジェクト・グループ内のすべてのレプリケート・オブジェクトが、そのスキーマから削除されます。

例外

表 34-78 DROP\_SNAPSHOT\_REPGROUP プロシージャの例外

例外	説明
nonsnapshot	起動サイトがスナップショット・サイトではありません。
missingrepgroup	指定したオブジェクト・グループが存在しません。

使用上の注意

DROP\_SNAPSHOT\_REPGROUP は、UNREGISTER\_SNAPSHOT\_REPGROUP を自動的にコールしてスナップショットの登録を解除します。ただし、登録解除中にエラーが発生しても無視します。

DROP\_SNAPSHOT\_REPOBJECT プロシージャ

このプロシージャは、スナップショット・サイトからレプリケート・オブジェクトを削除します。

構文

```
DBMS_REPCAT.DROP_SNAPSHOT_REPOBJECT (  
    sname          IN   VARCHAR2,  
    oname          IN   VARCHAR2,  
    type           IN   VARCHAR2,  
    drop_objects   IN   BOOLEAN := FALSE);
```

## パラメータ

表 34-79 DROP\_SNAPSHOT\_REPOBJECT プロシージャのパラメータ

パラメータ	説明
sname	オブジェクトが置かれているスキーマの名前。
oname	レプリケート・オブジェクト・グループから削除するオブジェクトの名前。
type	削除するオブジェクトの型。
drop_objects	デフォルトでは、オブジェクトは関連スキーマ内に残りますが、その関連オブジェクト・グループからは削除されます。現行スナップショット・サイトのスキーマからオブジェクトを完全に削除するには、この引数に TRUE を設定します。

## 例外

表 34-80 DROP\_SNAPSHOT\_REPOBJECT プロシージャの例外

例外	説明
nonsnapshot	起動サイトがスナップショット・サイトではありません。
missingobject	指定したオブジェクトが存在しません。
typefailure	指定した型パラメータはサポートされていません。

## DROP\_conflicttype\_RESOLUTION プロシージャ

このプロシージャは、更新、削除または一意性の競合解消ルーチンを削除します。このプロシージャは、マスター定義サイトからコールする必要があります。コールする必要があるプロシージャは、ルーチンが解消する競合の型によって決まります。

競合の型	プロシージャ名
更新	DROP_UPDATE_RESOLUTION
一意性	DROP_UNIQUE_RESOLUTION
削除	DROP_DELETE_RESOLUTION

構文

```
DBMS_REPCAT.DROP_UPDATE_RESOLUTION (  
    sname                IN   VARCHAR2,  
    oname                IN   VARCHAR2,  
    column_group         IN   VARCHAR2,  
    sequence_no          IN   NUMBER);
```

```
DBMS_REPCAT.DROP_DELETE_RESOLUTION (  
    sname                IN   VARCHAR2,  
    oname                IN   VARCHAR2,  
    sequence_no          IN   NUMBER);
```

```
DBMS_REPCAT.DROP_UNIQUE_RESOLUTION (  
    sname                IN   VARCHAR2,  
    oname                IN   VARCHAR2,  
    constraint_name      IN   VARCHAR2,  
    sequence_no          IN   NUMBER);
```

パラメータ

表 34-81 DROP\_conflicttype\_RESOLUTION プロシージャのパラメータ

パラメータ	説明
sname	表が置かれているスキーマ。
oname	競合解消ルーチンを削除する表の名前。
column_group	更新競合解消ルーチンを削除する列グループの名前。
constraint_name	一意性競合解消ルーチンを削除する一意性制約の名前。
sequence_no	削除する競合解消方法に割り当てられている順序番号。この番号でルーチンが一意に識別されます。

例外

表 34-82 DROP\_conflicttype\_RESOLUTION プロシージャの例外

例外	説明
nonmasterdef	起動サイトがマスター定義サイトではありません。
missingobject	指定したオブジェクトが、指定のスキーマ内の表として存在していないか、または指定した順序番号の競合解消ルーチンが登録されていません。
notquiesced	レプリケート・オブジェクト・グループが休止状態ではありません。

## EXECUTE\_DDL プロシージャ

このプロシージャは、一部またはすべてのマスター・サイトで実行される DDL を提供します。このプロシージャは、マスター定義サイトからコールする必要があります。

### 構文

```
DBMS_REPCAT.EXECUTE_DDL (
    gname          IN   VARCHAR2,
    { master_list  IN   VARCHAR2      := NULL
    | master_table IN   DBMS_UTILITY.DBLINK_ARRAY, }
    ddl_text       IN   VARCHAR2);
```

### パラメータ

表 34-83 EXECUTE\_DDL プロシージャのパラメータ

パラメータ	説明
gname	レプリケート・オブジェクト・グループの名前。
master_list	提供した DDL を実行するマスター・サイトの名前のカンマで区切られたリスト。サイト名の間に空白を挿入しないでください。デフォルト値 NULL は、マスター定義サイトも含め、すべてのサイトで DDL が実行されることを示します。
master_table	提供した DDL を実行するマスター・サイトの表。最初のマスターはオフセット 1 の位置、2 番目はオフセット 2 の位置、以下同様に設定されている必要があります。
ddl_text	指定した各マスター・サイトで実行する DDL。

### 例外

表 34-84 EXECUTE\_DDL プロシージャの例外

例外	説明
nonmasterdef	起動サイトがマスター定義サイトではありません。
nonmaster	マスター・サイト以外のサイトが少なくとも 1 つあります。
ddlfailure	マスター定義サイトで DDL が成功しませんでした。
commfailure	アクセスできないマスター・サイトが少なくとも 1 つあります。

使用上の注意

スキーマを指定せずに DDL を発行すると、レプリケーション管理者のスキーマがデフォルトのスキーマとして使用されます。レプリケーション管理者のスキーマを使用しない場合は、スキーマを必ず指定してください。このプロシージャはオーバーロードされています。MASTER\_LIST パラメータと MASTER\_TABLE パラメータは、両方同時には指定できません。

GENERATE\_REPLICATION\_SUPPORT プロシージャ

このプロシージャは、レプリケーションのサポートに必要なトリガーとパッケージを生成します。このプロシージャは、マスター定義サイトからコールする必要があります。

構文

```
DBMS_REPCAT.GENERATE_REPLICATION_SUPPORT (
  sname          IN    VARCHAR2,
  oname          IN    VARCHAR2,
  type           IN    VARCHAR2,
  package_prefix IN    VARCHAR2  := NULL,
  procedure_prefix IN  VARCHAR2  := NULL,
  distributed    IN    BOOLEAN   := TRUE,
  gen_objs_owner IN    VARCHAR2  := NULL,
  min_communication IN  BOOLEAN   := TRUE,
  generate_80_compatible IN  BOOLEAN := TRUE);
```

パラメータ

表 34-85 GENERATE\_REPLICATION\_SUPPORT プロシージャのパラメータ

パラメータ	説明
sname	オブジェクトが置かれているスキーマ。
oname	レプリケーション・サポートを生成するオブジェクトの名前。
type	オブジェクトの型。サポートされている型は、TABLE、PACKAGE および PACKAGE BODY です。
package_prefix	型が PACKAGE または PACKAGE BODY のオブジェクトの場合は、生成されたラッパー・パッケージ名の前にこのパラメータの値が付加されます。デフォルトは DEFER_ です。
procedure_prefix	型が PACKAGE または PACKAGE BODY のオブジェクトの場合は、生成されたラッパー・プロシージャ名の前にこのパラメータの値が付加されます。デフォルトでは、接頭辞は割り当てられません。
distributed	この値は TRUE に設定してください。

表 34-85 GENERATE\_REPLICATION\_SUPPORT プロシージャのパラメータ

パラメータ	説明
gen_objs_owner	型が PACKAGE または PACKAGE BODY のオブジェクトの場合は、生成されたオブジェクトが作成されるスキーマを指定します。NULL の場合、オブジェクトは sname に作成されます。
min_communication	マスター・サイトのいずれかが Oracle7 リリース 7.3 を実行している場合は、FALSE を設定します。新旧の値の伝播を最小化する場合は、TRUE を設定します。デフォルトは TRUE です。詳細は、『Oracle8i レプリケーション・ガイド』の「競合の解消」を参照してください。
generate_80_compatible	マスター・サイトのいずれかが Oracle8i リリース 8.1.5 以前のバージョンの Oracle Server を実行している場合は、TRUE を設定します。レプリケート環境が純粋な Oracle8i リリース 8.1.5 以上の環境の場合は、FALSE を設定します。

## 例外

表 34-86 GENERATE\_REPLICATION\_SUPPORT プロシージャの例外

例外	説明
nomasterdef	起動サイトがマスター定義サイトではありません。
missingobject	指定したオブジェクトが、行レベル・レプリケーション情報を待機している表として指定のスキーマ内に存在していないか、またはラッパー生成を待機しているパッケージ（本体）として存在していません。
typefailure	指定した型パラメータはサポートされていません。
notquiesced	レプリケート・オブジェクト・グループが中断されていません。
commfailure	アクセスできないマスター・サイトが少なくとも 1 つあります。
missingschema	スキーマが存在しません。
dbnotcompatible	マスターの 1 つがリリース 7.3 と互換性がありません。
duplicateobject	オブジェクトがすでに存在します。

## GENERATE\_SNAPSHOT\_SUPPORT プロシージャ

このプロシージャは、トリガーを起動し、更新可能スナップショットのレプリケーションまたはプロシージャ・レプリケーションのサポートに必要なパッケージを生成します。このプロシージャは、スナップショット・サイトからコールする必要があります。

構文

```
DBMS_REPCAT.GENERATE_SNAPSHOT_SUPPORT (
    sname                IN VARCHAR2,
    oname                IN VARCHAR2,
    type                 IN VARCHAR2,
    gen_objs_owner       IN VARCHAR2 := '',
    min_communication    IN BOOLEAN  := TRUE,
    generate_80_compatible IN BOOLEAN := TRUE);
```

パラメータ

表 34-87 GENERATE\_SNAPSHOT\_SUPPORT プロシージャのパラメータ

パラメータ	説明
sname	オブジェクトが置かれているスキーマ。
oname	サポートを生成するオブジェクトの名前。
type	オブジェクトの型。サポートされている型は、SNAPSHOT、PACKAGE および PACKAGE BODY です。
gen_objs_owner	型が PACKAGE または PACKAGE BODY のオブジェクトの場合は、生成されたオブジェクトが作成されるスキーマを指定します。NULL の場合、オブジェクトは sname に作成されます。
min_communication	TRUE を設定すると、更新文で列が変更された場合、更新トリガーはその列の新しい値のみ送信します。その列がキー列または変更された列グループ内の列の場合、更新トリガーはその列の元の値のみ送信します。
generate_80_compatible	マスター・サイトのいずれかが Oracle8i リリース 8.1.5 以前のバージョンの Oracle Server を実行している場合は、TRUE を設定します。レプリケート環境が純粋な Oracle8i リリース 8.1.5 以上の環境の場合は、FALSE を設定します。

例外

表 34-88 GENERATE\_SNAPSHOT\_SUPPORT プロシージャの例外

例外	説明
nonsnapshot	起動サイトがスナップショット・サイトではありません。
missingobject	指定したオブジェクトが、行または列レベルのレプリケーション情報を待機しているスナップショットとしてレプリケート・スキーマ内に存在していないか、またはラッパー生成を待機しているパッケージ（本体）として存在していません。



表 34-88 GENERATE\_SNAPSHOT\_SUPPORT プロシージャの例外

例外	説明
typefailure	指定した型パラメータはサポートされていません。
missingschema	生成オブジェクトの指定の所有者が存在しません。
missingremoteobject	マスター・オブジェクトが、レプリケーション・サポートをまだ生成していません。
commfailure	マスターにアクセスできません。

## 使用上の注意

CREATE\_SNAPSHOT\_REPOBJECT は、更新可能スナップショットのスナップショット・サポートを自動的に生成します。

## MAKE\_COLUMN\_GROUP プロシージャ

このプロシージャは、1 つ以上のメンバーを含んだ新しい列グループを作成します。このプロシージャは、マスター定義サイトからコールする必要があります。

**関連項目：** 詳細は、『Oracle8i レプリケーション・ガイド』の「競合の解消」を参照してください。

## 構文

```
DBMS_REPCAT.MAKE_COLUMN_GROUP (
    sname          IN   VARCHAR2,
    oname          IN   VARCHAR2,
    column_group   IN   VARCHAR2,
    list_of_column_names IN VARCHAR2 | DBMS_REPCAT.VARCHAR2S);
```

## パラメータ

表 34-89 MAKE\_COLUMN\_GROUP プロシージャのパラメータ

パラメータ	説明
sname	レプリケート表が置かれているスキーマ。
oname	新しい列グループを作成するレプリケート表の名前。
column_group	作成する列グループに割り当てる名前。
list_of_column_names	グループ化する列の名前。列名は、名前のカンマで区切られたリストまたは PL/SQL 表のいずれかで示します。PL/SQL 表は dbms_repcat.varchar2 型にしてください。表内のすべての列を含んだ列グループを作成するには、単一の値 '*' を使用します。

例外

表 34-90 MAKE\_COLUMN\_GROUP プロシージャの例外

例外	説明
nonmasterdef	起動サイトがマスター定義サイトではありません。
duplicategroup	指定した列グループが、表にすでに存在しています。
missingobject	指定した表が存在しません。
missingcolumn	指定した列が、指定した表内に存在していません。
duplicatecolumn	指定した列は、すでに別の列グループのメンバーです。
notquiesced	レプリケート・オブジェクト・グループが休止状態ではありません。

PURGE\_MASTER\_LOG プロシージャ

このプロシージャは、指定した識別番号、発信元またはレプリケート・オブジェクト・グループに対応する RepCatLog 内のローカル・メッセージを削除します。

構文

```
DBMS_REPCAT.PURGE_MASTER_LOG (  
    id      IN    NATURAL,  
    source  IN    VARCHAR2,  
    gname   IN    VARCHAR2);
```

パラメータ

表 34-91 PURGE\_MASTER\_LOG プロシージャのパラメータ

パラメータ	説明
id	要求の識別番号。RepCatLog ビューに表示される番号です。
source	要求発信元のマスター・サイト。
gname	要求の作成対象となったレプリケート・オブジェクト・グループの名前。

例外

表 34-92 PURGE\_MASTER\_LOG プロシージャの例外

例外	説明
nonmaster	gname が NULL でなく、起動サイトがマスター・サイトではありません。

PURGE\_STATISTICS プロシージャ

このプロシージャは、RepResolution\_Statistics ビューから情報を削除します。

構文

```
DBMS_REPCAT.PURGE_STATISTICS (  
    sname      IN   VARCHAR2,  
    oname      IN   VARCHAR2,  
    start_date IN   DATE,  
    end_date   IN   DATE);
```

パラメータ

表 34-93 PURGE\_STATISTICS プロシージャのパラメータ

パラメータ	説明
sname	レプリケート表が置かれているスキーマの名前。
oname	競合解消統計を削除する表の名前。
start_date/end_date	統計を削除する日付範囲。START_DATE が NULL の場合は、END_DATE までの統計がすべて削除されます。END_DATE が NULL の場合は、START_DATE 以降の統計がすべて削除されます。

例外

表 34-94 PURGE\_STATISTICS プロシージャの例外

例外	説明
missingschema	指定したスキーマが存在しません。
missingobject	指定した表が存在しません。
statnotreg	この表は統計収集をするために登録されていません。

## REFRESH\_SNAPSHOT\_REPGROUP プロシージャ

このプロシージャは、スナップショット・サイトのオブジェクト・グループを、関連するマスター・サイトから最新データでリフレッシュします。

### 構文

```
DBMS_REPCAT.REFRESH_SNAPSHOT_REPGROUP (  
    gname                IN    VARCHAR2,  
    drop_missing_contents IN    BOOLEAN    := FALSE,  
    refresh_snapshots    IN    BOOLEAN    := FALSE,  
    refresh_other_objects IN    BOOLEAN    := FALSE);
```

### パラメータ

表 34-95 REFRESH\_SNAPSHOT\_REPGROUP プロシージャのパラメータ

パラメータ	説明
gname	レプリケート・オブジェクト・グループの名前。
drop_missing_contents	オブジェクトがレプリケート・オブジェクト・グループから削除された場合、スナップショット・サイトのスキーマからは自動的に削除されません。ただし、オブジェクトはレプリケートされなくなります。つまり、このオブジェクトに対する変更は、関連するマスター・サイトに送信されなくなります。スナップショットは、引き続き関連するマスター表からリフレッシュされますが、更新可能スナップショットに対する変更は失われます。この引数に TRUE を設定すると、オブジェクト・グループからオブジェクトが削除されるときに、そのオブジェクトをスキーマから完全に削除できます。
refresh_snapshots	これを TRUE に設定すると、レプリケート・オブジェクト・グループ内のスナップショットの内容がリフレッシュされます。
refresh_other_objects	これを TRUE に設定すると、レプリケート・オブジェクト・グループ内の非スナップショット・オブジェクトの内容がリフレッシュされます。

## 例外

表 34-96 REFRESH\_SNAPSHOT\_REPGROUP プロシージャの例外

例外	説明
nonsnapshot	起動サイトがスナップショット・サイトではありません。
nonmaster	このマスターは、マスター・サイトではありません。
commfailure	マスターにアクセスできません。
missingrepgroup	オブジェクト・グループ名が指定されていません。

## REGISTER\_SNAPSHOT\_REPGROUP プロシージャ

このプロシージャは、registered\_snapshot\_groups の挿入、変更および削除を行うことによって、それぞれのマスター・サイトのスナップショット管理を容易にします。

## 構文

```
DBMS_REPCAT.REGISTER_SNAPSHOT_REPGROUP (
    gname          IN   VARCHAR2,
    snapsite       IN   VARCHAR2,
    comment        IN   VARCHAR2  := NULL,
    rep_type       IN   NUMBER     := reg_unknown,
    fname          IN   VARCHAR2  := NULL);
```

## パラメータ

表 34-97 REGISTER\_SNAPSHOT\_REPGROUP プロシージャのパラメータ

パラメータ	説明
gname	登録するスナップショット・オブジェクト・グループの名前。
snapsite	スナップショット・サイトのグローバル名。
comment	スナップショット・サイトに対するコメント、または既存のコメントに対する更新。
rep_type	スナップショット・グループのバージョン。代入できる有効な定数は、reg_unknown (デフォルト)、reg_v7_group、reg_v8_group および reg_repapi_group です。
fname	内部使用のためのシステム・パラメータ。オラクル社カスタマ・サポートから指示がない限り、このパラメータは設定しないでください。

例外

表 34-98 REGISTER\_SNAPSHOT\_REPGROUP プロシージャの例外

例外	説明
missingrepgroup	オブジェクト・グループ名が指定されていません。
nullsitename	スナップショット・サイトが指定されていません。
nonmaster	このプロシージャは、スナップショットのマスター・サイトで実行する必要があります。
duplicaterepgroup	オブジェクトがすでに存在します。

REGISTER\_STATISTICS プロシージャ

このプロシージャは、表の更新競合、削除競合および一意性競合の正常な解消に関する情報を収集します。

構文

```
DBMS_REPCAT.REGISTER_STATISTICS (  
    sname IN   VARCHAR2,  
    oname IN   VARCHAR2);
```

パラメータ

表 34-99 REGISTER\_STATISTICS プロシージャのパラメータ

パラメータ	説明
sname	表が置かれているスキーマの名前。
oname	競合解消統計を収集する表の名前。

例外

表 34-100 REGISTER\_STATISTICS プロシージャの例外

例外	説明
missingschema	指定したスキーマが存在しません。
missingobject	指定した表が存在しません。

## RELOCATE\_MASTERDEF プロシージャ

このプロシージャは、マスター定義サイトをレプリケート環境内の別のマスター・サイトに変更します。

### 構文

```
DBMS_REPCAT.RELOCATE_MASTERDEF (  
    gname                IN   VARCHAR2,  
    old_masterdef         IN   VARCHAR2,  
    new_masterdef         IN   VARCHAR2,  
    notify_masters        IN   BOOLEAN    := TRUE,  
    include_old_masterdef IN   BOOLEAN    := TRUE,  
    require_flavor_change IN   BOOLEAN    := FALSE);
```

### パラメータ

表 34-101 RELOCATE\_MASTERDEF プロシージャのパラメータ

パラメータ	説明
gname	マスター定義を再配置するオブジェクト・グループの名前。
old_masterdef	現行のマスター定義サイトの完全修飾データベース名。
new_masterdef	新しいマスター定義サイトにする既存のマスター・サイトの完全修飾データベース名。
notify_masters	TRUE の場合、このプロシージャは、変更内容をすべてのマスターに同期式でマルチキャストします ( INCLUDE_OLD_MASTERDEF が TRUE の場合のみ OLD_MASTERDEF も含まれます )。変更を適用できないマスターがある場合は、すべてのマスターで変更がロールバックされます。
include_old_masterdef	NOTIFY_MASTERS が TRUE で、INCLUDE_OLD_MASTERDEF も TRUE の場合は、元のマスター定義サイトにも変更内容が通知されます。
require_flavor_change	内部使用のためのシステム・パラメータ。オラクル社カスタマ・サポートから指示がない限り、このパラメータは設定しないでください。

例外

表 34-102 RELOCATE\_MASTERDEF プロシージャの例外

例外	説明
nonmaster	NEW_MASTERDEF がマスター・サイトではないか、または起動サイトがマスター・サイトではありません。
nonmasterdef	OLD_MASTERDEF がマスター定義サイトではありません。
connfailure	NOTIFY_MASTERS が TRUE ですが、アクセスできないマスター・サイトが少なくとも 1 つあります。

使用上の注意

RELOCATE\_MASTERDEF のコール時に、元のマスター定義サイトと新しいマスター定義サイトのいずれも使用可能である必要はありません。計画的に再構成する場合は、NOTIFY\_MASTERS に TRUE および INCLUDE\_OLD\_MASTERDEF に TRUE を設定して RELOCATE\_MASTERDEF を起動してください。

マスター定義サイトでのみ障害が発生した場合は、NOTIFY\_MASTERS に TRUE および INCLUDE\_OLD\_MASTERDEF に FALSE を設定して RELOCATE\_MASTERDEF を起動してください。複数のマスター・サイトとマスター定義サイトで障害が発生した場合、管理者は、NOTIFY\_MASTERS に FALSE を設定して、各マスターで RELOCATE\_MASTERDEF を起動する必要があります。

REMOVE\_MASTER\_DATABASES プロシージャ

このプロシージャは、レプリケート環境から 1 つ以上のマスター・データベースを削除します。また、トリガーと関連するパッケージを、残りのマスター・サイトで再生成します。このプロシージャは、マスター定義サイトからコールする必要があります。

構文

```
DBMS_REPCAT.REMOVE_MASTER_DATABASES (  
  gname          IN   VARCHAR2,  
  master_list     IN   VARCHAR2 |  
  master_table    IN   DBMS_UTILITY.DBLINK_ARRAY);
```



## パラメータ

**表 34-103 REMOVE\_MASTER\_DATABASES プロシージャのパラメータ**

パラメータ	説明
gname	レプリケート環境に関連付けられているオブジェクト・グループの名前。マスター・データベースが複数のレプリケート環境で使用されている場合に、このパラメータによって混乱を避けることができます。
master_list	レプリケート環境から削除する完全修飾マスター・データベース名のカンマで区切られたリスト。リストの名前の間に空白を挿入しないでください。
master_table	リストのかわりに、型 DBMS_UTILITY.DBLINK_ARRAY の PL/SQL 表でデータベース名を指定することもできます。

## 例外

**表 34-104 REMOVE\_MASTER\_DATABASES プロシージャの例外**

例外	説明
nonmasterdef	起動サイトがマスター定義サイトではありません。
nonmaster	指定したデータベースの中に、マスター・サイト以外のデータベースが少なくとも 1 つあります。
reconfigerror	指定したデータベースの 1 つが、マスター定義サイトです。
commfailure	残りのマスター・サイトの中に、アクセスできないサイトが少なくとも 1 つあります。

## REPCAT\_IMPORT\_CHECK プロシージャ

このプロシージャは、レプリケート・オブジェクトまたはアドバンスド・レプリケーション機能で使用するオブジェクトのエクスポートまたはインポートの実行後、レプリケート・オブジェクト・グループ内のオブジェクトの識別子と状態値が適切かどうかを確認します。

## 構文

```
DBMS_REPCAT.REPCAT_IMPORT_CHECK (
    gname      IN   VARCHAR2,
    master     IN   BOOLEAN);
```

パラメータ

表 34-105 REPCAT\_IMPORT\_CHECK プロシージャのパラメータ

パラメータ	説明
gname	レプリケート・オブジェクト・グループの名前。パラメータを両方とも省略すると、現行サイトのすべてのレプリケート・オブジェクト・グループがチェックされます。
master	マスター・サイトをチェックする場合は TRUE、スナップショット・サイトをチェックする場合は FALSE を設定します。

例外

表 34-106 REPCAT\_IMPORT\_CHECK プロシージャの例外

例外	説明
nonmaster	MASTER が TRUE ですが、データベースがそのオブジェクト・グループのマスター・サイトではないか、または該当のデータベースではありません。
nonsnapshot	MASTER が FALSE ですが、データベースがそのオブジェクト・グループのスナップショット・サイトではありません。
missingobject	オブジェクト・グループ内に有効なレプリケート・オブジェクトが存在しません。
missingrepgroup	指定したグループ名が存在しません。

RESUME\_MASTER\_ACTIVITY プロシージャ

このプロシージャは、レプリケート環境の休止後、通常のレプリケーション・アクティビティを再開します。

構文

```
DBMS_REPCAT.RESUME_MASTER_ACTIVITY (  
  gname      IN  VARCHAR2,  
  override   IN  BOOLEAN := FALSE);
```

## パラメータ

表 34-107 RESUME\_MASTER\_ACTIVITY プロシージャのパラメータ

パラメータ	説明
gname	レプリケート・オブジェクト・グループの名前。
override	TRUE の場合、保留中の RepCat 管理要求は無視され、通常のレプリケーション・アクティビティが各マスターで可能な限り迅速に再開されます。この設定は、緊急の場合に限り使用してください。  FALSE の場合は、各マスターの gname に対する保留中の RepCat 管理要求がない場合に限り、そのマスターで通常のレプリケーション・アクティビティが再開されます。

## 例外

表 34-108 RESUME\_MASTER\_ACTIVITY プロシージャの例外

例外	説明
nomasterdef	起動サイトがマスター定義サイトではありません。
notquiesced	レプリケート・オブジェクト・グループが休止中または休止状態ではありません。
commfailure	アクセスできないマスター・サイトが少なくとも 1 つあります。

## SEND\_OLD\_VALUES プロシージャ

更新と削除のために、レプリケート表の各非キー列の元の値を送信するオプションがあります。デフォルトでは、すべての列の元の値が送信されます。DBMS\_REPCAT.SEND\_OLD\_VALUES をマスター定義サイトで起動すると、すべてのマスター・サイトとスナップショット・サイトでこの動作を変更できます。

## 構文

```
DBMS_REPCAT.SEND_OLD_VALUES(
    sname          IN  VARCHAR2,
    oname          IN  VARCHAR2,
    { column_list  IN  VARCHAR2
    | column_table IN  DBMS_REPCAT.VARCHAR2s, }
    operation      IN  VARCHAR2 := 'UPDATE',
    send           IN  BOOLEAN  := TRUE );
```

**注意：** このプロシージャはオーバーロードされています。column\_list パラメータと column\_table パラメータは、両方同時には指定できません。

パラメータ

表 34-109 SEND\_OLD\_VALUES プロシージャのパラメータ

パラメータ	説明
sname	表が置かれているスキーマ。
oname	レプリケート表の名前。
column_list	表内の列のカンマで区切られたリスト。エントリ間に空白を挿入しないでください。
column_table	リストのかわりに、列名を含んだ DBMS_REPCAT.VARCHAR2 型の PL/SQL 表を使用できます。最初の列名はオフセット 1 の位置、2 番目はオフセット 2 の位置、以下同様に設定されている必要があります。
operation	有効な値は、UPDATE、DELETE、またはアスタリスクのワイルドカード '*' (更新と削除を意味します) です。
send	TRUE の場合は、指定した列の元の値が送信されます。FALSE の場合は、指定した列の元の値が送信されません。指定外の列と指定外の操作には影響しません。マスター定義サイトでは、表の min_communication が TRUE になると、指定した変更がすぐに有効となります。変更内容がマスター・サイトまたはスナップショット・サイトで有効になるのは、min_communication に TRUE を設定して、次回そのサイトでレプリケーション・サポートを生成したときです。

---

**注意：** operation パラメータを使用すると、行の削除時または非キー列の更新時に、非キー列の元の値を送信するかどうかを決定できます。元の値を送信しないと、元の値のかわりに NULL が送信され、更新または削除の適用時に送信先の現在の列値と元の値が等しいとみなされます。

Oracle のデフォルト動作を変更する前に、『Oracle8i レプリケーション・ガイド』の「更新の競合解消のためのデータ伝播の最少化」を参照してください。

---

## 例外

**表 34-110 SEND\_OLD\_VALUES プロシージャの例外**

例外	説明
nomasterdef	起動サイトがマスター定義サイトではありません。
missingobject	指定したオブジェクトは、行レベル・レプリケーション情報を待機している表として指定のスキーマ内に存在していません。
missingcolumn	表に存在しない列が少なくとも 1 つあります。
notquiesced	レプリケート・オブジェクト・グループが中断されていません。
typefailure	無効な操作が指定されています。

## SET\_COLUMNS プロシージャ

このプロシージャは、行レベル・レプリケーションの使用時に、主キーのかわりに代替列または列グループを使用するために、表のどの列を比較するかを判断します。このプロシージャは、マスター定義サイトからコールする必要があります。

**関連項目：** 『Oracle8i レプリケーション・ガイド』の「マルチマスタ・レプリケーションの使用方法」を参照してください。

## 構文

```
DBMS_REPCAT.SET_COLUMNS (
    sname          IN    VARCHAR2,
    oname          IN    VARCHAR2,
    { column_list  IN    VARCHAR2
    | column_table IN    DBMS_UTILITY.NAME_ARRAY } );
```

**注意：** このプロシージャはオーバーロードされています。column\_list パラメータと column\_table パラメータは、両方同時には指定できません。

パラメータ

表 34-111 SET\_COLUMNS プロシージャのパラメータ

パラメータ	説明
sname	表が置かれているスキーマ。
oname	表の名前。
column_list	表の中で主キーとして使用する列のカンマで区切られたリスト。エントリ間に空白を挿入しないでください。
column_table	リストのかわりに、列名を含んだ型 DBMS_UTILITY.NAME_ARRAY の PL/SQL 表を使用できます。最初の列名はオフセット 1 の位置、2 番目はオフセット 2 の位置、以下同様に設定されている必要があります。

例外

表 34-112 SET\_COLUMNS プロシージャの例外

例外	説明
nonmasterdef	起動サイトがマスター定義サイトではありません。
missingobject	指定したオブジェクトは、行レベル・レプリケーション情報を待機している表として指定のスキーマ内に存在していません。
missingcolumn	表に存在しない列が少なくとも 1 つあります。

SUSPEND\_MASTER\_ACTIVITY プロシージャ

このプロシージャは、オブジェクト・グループのレプリケーション・アクティビティを中断します。このプロシージャは、マスター定義サイトからコールする必要があります。

構文

```
DBMS_REPCAT.SUSPEND_MASTER_ACTIVITY (  
    gname    IN    VARCHAR2);
```

## パラメータ

表 34-113 SUSPEND\_MASTER\_ACTIVITY プロシージャのパラメータ

パラメータ	説明
gname	アクティビティを中断するオブジェクト・グループの名前。

## 例外

表 34-114 SUSPEND\_MASTER\_ACTIVITY プロシージャの例外

例外	説明
nomasterdef	起動サイトがマスター定義サイトではありません。
notnormal	レプリケート・オブジェクト・グループが通常の操作モードではありません。
commfailure	アクセスできないマスター・サイトが少なくとも 1 つあります。

## 使用上の注意

現行の SUSPEND\_MASTER\_ACTIVITY のインプリメンテーションでは、各マスター・サイトのレプリケート・オブジェクト・グループをすべて休止します。

## SWITCH\_SNAPSHOT\_MASTER プロシージャ

このプロシージャは、スナップショット・レプリケート・オブジェクト・グループのマスター・データベースを別のマスター・サイトに変更します。影響するスナップショットは完全にリフレッシュされ、必要に応じて、トリガーと関連するパッケージが再生成されます。このプロシージャは、マスターの変更前に、元のマスター・サイトにキューを送信しません。

## 構文

```
DBMS_REPCAT.SWITCH_SNAPSHOT_MASTER (  
    gname          IN   VARCHAR2,  
    master         IN   VARCHAR2);
```

パラメータ

表 34-115 SWITCH\_SNAPSHOT\_MASTER プロシージャのパラメータ

パラメータ	説明
gname	マスター・サイトを変更するスナップショット・オブジェクト・グループの名前。
master	そのスナップショット・サイトに使用する新しいマスター・データベースの完全修飾データベース名。

例外

表 34-116 SWITCH\_SNAPSHOT\_MASTER プロシージャの例外

例外	説明
nonsnapshot	起動サイトがスナップショット・サイトではありません。
nonmaster	指定したデータベースはマスター・サイトではありません。
commfailure	指定したデータベースにアクセスできません。

UNREGISTER\_SNAPSHOT\_REPGROUP プロシージャ

このプロシージャは、registered\_snapshot\_groups の挿入、変更および削除を行うことによって、それぞれのマスター・サイトのスナップショット管理を容易にします。

構文

```
DBMS_REPCAT.UNREGISTER_SNAPSHOT_REPGROUP (  
    gname      IN   VARCHAR2,  
    snapsite   IN   VARCHAR2);
```

パラメータ

表 34-117 UNREGISTER\_SNAPSHOT\_REPGROUP プロシージャのパラメータ

パラメータ	説明
gname	登録解除するスナップショット・オブジェクト・グループの名前。
snapsite	スナップショット・サイトのグローバル名。



## VALIDATE ファンクション

このファンクションは、複数マスター・レプリケーション環境のキー状態が正しいかどうかを検証します。このプロシージャはオーバーロードされています。

### 構文

```
DBMS_REPCAT.VALIDATE (  
    gname                IN  VARCHAR2,  
    check_genflags       IN  BOOLEAN := FALSE,  
    check_valid_objs     IN  BOOLEAN := FALSE,  
    check_links_sched    IN  BOOLEAN := FALSE,  
    check_links          IN  BOOLEAN := FALSE,  
    error_table          OUT dbms_repcat.validate_err_table )  
RETURN BINARY_INTEGER;
```

```
DBMS_REPCAT.VALIDATE (  
    gname                IN  VARCHAR2,  
    check_genflags       IN  BOOLEAN := FALSE,  
    check_valid_objs     IN  BOOLEAN := FALSE,  
    check_links_sched    IN  BOOLEAN := FALSE,  
    check_links          IN  BOOLEAN := FALSE,  
    error_msg_table      OUT DBMS_UTILITY.UNCL_ARRAY,  
    error_num_table      OUT DBMS_UTILITY.NUMBER_ARRAY )  
RETURN BINARY_INTEGER;
```

### パラメータ

表 34-118 VALIDATE ファンクションのパラメータ

パラメータ	説明
gname	検証するマスター・グループの名前。
check_genflags	グループ内のオブジェクトがすべて生成済みであるかどうかをチェックします。このチェックは、マスター定義サイトでのみ実行してください。
check_valid_objs	グループ内のオブジェクトに対応する基礎オブジェクトが有効かどうかをチェックします。このチェックは、マスター定義サイトでのみ実行してください。マスター定義サイトは他のすべてのサイトを調べて、基礎オブジェクトが有効であることを確認します。オブジェクトの有効性は、接続ユーザーのスキーマ内でチェックされます。
check_links_sched	リンクの実行がスケジュールされているかどうかをチェックします。このチェックは、各マスター・サイトで起動してください。

表 34-118 VALIDATE ファンクションのパラメータ

パラメータ	説明
check_links	接続ユーザー（レプリケーション管理者）と伝播担当者に、レプリケーションが適切に動作するための正しいリンクがあるかどうかをチェックします。リンクがデータベースに存在していて、アクセス可能であることを確認します。このチェックは、各マスター・サイトで起動してください。
error_table	検出されたすべてのエラーのメッセージと番号を戻します。
error_msg_table	検出されたすべてのエラーのメッセージを戻します。
error_num_table	検出されたすべてのエラーの番号を戻します。

例外

表 34-119 VALIDATE ファンクションの例外

例外	説明
missingdblink	データベース・リンクがレプリケーション伝播担当者のスキーマ内に存在していないか、またはスケジュールされていません。データベース・リンクがデータベースに存在していてアクセス可能であること、および実行がスケジュールされていることを確認してください。
dblinkmismatch	ローカル・ノードのデータベース・リンク名が、そのリンクがアクセスするデータベースのグローバル名と一致しません。グローバル名に TRUE が設定され、そのリンク名がグローバル名と一致していることを確認してください。
dblinkuidmismatch	ローカル・ノードのレプリケーション管理ユーザーのユーザー名と、そのデータベース・リンクに対応するノードのユーザー名が同じではありません。アドバンスト・レプリケーションでは、両方のユーザーが同じであることが必要です。ローカル・ノードのレプリケーション管理ユーザーのユーザー ID と、そのデータベース・リンクに対応するノードのユーザー ID が同じになるようにしてください。
objectnotgenerated	オブジェクトが、他のマスター・サイトで生成されていないか、または生成中です。マスター定義サイトのオブジェクトに対して generate_replication_support と do_deferred_repcat_admin をコールして、オブジェクトを確実に生成してください。
opnotsupported	オブジェクト・グループがバージョン 8 より前のノードでレプリケートされている場合は、操作がサポートされていません。レプリケート・オブジェクト・グループのすべてのノードがバージョン 8 であることを確認してください。

使用上の注意

VALIDATE の戻り値は、検出されたエラーの数です。ファンクションの OUT パラメータには、検出されたエラーが戻されます。最初のインタフェース・ファンクションでは、ERROR\_TABLE はレコードの配列で構成されています。各レコードには、VARCHAR2 と NUMBER があります。文字列フィールドにはエラー・メッセージが格納され、番号フィールドには Oracle のエラー番号が格納されます。

2 番目のインタフェースは、OUT 配列が 2 つある以外は最初のインタフェースと同じです。VARCHAR2 配列にエラー・メッセージが格納され、NUMBER 配列にエラー番号が格納されます。

WAIT\_MASTER\_LOG プロシージャ

このプロシージャは、マスター・サイトに非同期で伝播された変更内容が適用されたかどうかを判断します。

構文

```
DBMS_REPCAT.WAIT_MASTER_LOG (  
    gname          IN    VARCHAR2,  
    record_count   IN    NATURAL,  
    timeout        IN    NATURAL,  
    true_count     OUT   NATURAL);
```

パラメータ

表 34-120 WAIT\_MASTER\_LOG プロシージャのパラメータ

パラメータ	説明
gname	レプリケート・オブジェクト・グループの名前。
record_count	未完了のアクティビティ件数がこのしきい値以下になるたびにプロシージャに戻ります。
timeout	プロシージャが戻るまで待機する最大秒数。
true_count (out parameter)	未完了のアクティビティの数を戻します。

例外

表 34-121 WAIT\_MASTER\_LOG プロシージャの例外

例外	説明
nonmaster	起動サイトがマスター・サイトではありません。



---

## DBMS\_REPCAT\_ADMIN

DBMS\_REPCAT\_ADMIN によって、対称型レプリケーション機能に必要な権限を付与したユーザーを作成できます。

# サブプログラムの要約

表 35-1 DBMS\_REPCAT\_ADMIN パッケージのサブプログラム

サブプログラム	説明
<a href="#">GRANT_ADMIN_ANY_SCHEMA プロシージャ</a> (35-2 ページ)	現行サイトでレプリケート・オブジェクト・グループを管理するために必要な権限を、レプリケーション管理者に付与します。
<a href="#">GRANT_ADMIN_SCHEMA プロシージャ</a> (35-3 ページ)	現行サイトでスキーマを管理するために必要な権限を、レプリケーション管理者に付与します。
<a href="#">REGISTER_USER_REPGROUP プロシージャ</a> (35-3 ページ)	リモート・サイトで使用するための代理スナップショット管理者権限または受信者権限を、マスター・サイトで割り当てます。
<a href="#">REVOKE_ADMIN_ANY_SCHEMA プロシージャ</a> (35-5 ページ)	GRANT_ADMIN_ANY_SCHEMA によって付与された権限とロールを、レプリケーション管理者から取り消します。
<a href="#">REVOKE_ADMIN_SCHEMA プロシージャ</a> (35-6 ページ)	GRANT_ADMIN_SCHEMA によって付与された権限とロールを、レプリケーション管理者から取り消します。
<a href="#">UNREGISTER_USER_REPGROUP プロシージャ</a> (35-6 ページ)	REGISTER_USER_REPGROUP プロシージャによって付与された権限とロールを、代理スナップショット管理者または受信者から取り消します。

## GRANT\_ADMIN\_ANY\_SCHEMA プロシージャ

このプロシージャは、現行サイトでレプリケート・オブジェクト・グループを管理するために必要な権限を、レプリケーション管理者に付与します。

### 構文

```
DBMS_REPCAT_ADMIN.GRANT_ADMIN_ANY_SCHEMA (  
    username IN VARCHAR2);
```

### パラメータ

表 35-2 GRANT\_ADMIN\_ANY\_SCHEMA プロシージャのパラメータ

パラメータ	説明
username	現行サイトでレプリケート・オブジェクト・グループを管理するために必要な権限とロールを付与するレプリケーション管理者の名前。

例外

表 35-3 GRANT\_ADMIN\_ANY\_REPGROUP プロシージャの例外

例外	説明
ORA-01917	ユーザーが存在しません。

GRANT\_ADMIN\_SCHEMA プロシージャ

このプロシージャは、現行サイトでスキーマを管理するために必要な権限をレプリケーション管理者に付与します。このプロシージャは、オブジェクト・グループが単一のスキーマ内にある場合に最も役立ちます。

構文

```
DBMS_REPCAT_ADMIN.GRANT_ADMIN_SCHEMA (  
    username IN VARCHAR2);
```

パラメータ

表 35-4 GRANT\_ADMIN\_REPSchema プロシージャのパラメータ

パラメータ	説明
username	レプリケーション管理者の名前。このユーザーには、現行サイトでレプリケート・オブジェクト・グループにある同名のスキーマを管理するために必要な権限とロールが付与されます。

例外

表 35-5 GRANT\_ADMIN\_REPSchema プロシージャの例外

例外	説明
ORA-01917	ユーザーが存在しません。

REGISTER\_USER\_REPGROUP プロシージャ

このプロシージャは、リモート・サイトで使用するための代理スナップショット管理者権限または受信者権限を、マスター・サイトで割り当てます。このプロシージャは、GRANT\_ADMIN\_SCHEMA や GRANT\_ADMIN\_ANY\_SCHEMA プロシージャで付与される強力な権限は付与せずに、代理スナップショット管理者または受信者に必要な権限のみを付与します。

**関連項目：** 信頼性のあるセキュリティ・モデルと信頼性の低いセキュリティ・モデルの詳細は、『Oracle8i レプリケーション・ガイド』の「高度な設定」を参照してください。

構文

```
DBMS_REPCAT_ADMIN.REGISTER_USER_REPGROUP (  
  username          IN   VARCHAR2,  
  privilege_type     IN   VARCHAR2,  
  list_of_gnames     IN   VARCHAR2 |  
  table_of_gnames    IN   dbms_utility.name_array);
```

パラメータ

表 35-6 REGISTER\_USER\_REPGROUP プロシージャのパラメータ

パラメータ	説明
username	代理スナップショット管理者権限または受信者権限のいずれかを付与するユーザーの名前。
privilege_type	割り当てる権限タイプ。privilege_type の定義には次の値を使用します。  RECEIVER: 受信者権限  PROXY_SNAPADMIN: 代理スナップショット管理者権限
list_of_gnames	ユーザーの受信者権限を登録するオブジェクト・グループのカンマで区切られたリスト。リストのエントリ間に空白を挿入しないでください。NULL を設定すると、このプロシージャのコール時点で認識されていないオブジェクト・グループも含め、すべてのオブジェクト・グループに対してそのユーザーが登録されます。NULL を設定するには、名前表記法を使用する必要があります。リスト内に無効なオブジェクト・グループがあると、リスト全体の登録に失敗します。  list_of_gnames に値を指定した場合は、table_of_gnames の値は指定しないでください。
table_of_gnames	ユーザーの受信者権限を登録するオブジェクト・グループの PL/SQL 表。PL/SQL 表の型は DBMS_UTILITY.NAME_ARRAY にしてください。この表は 1 が基準です。値に NULL を使用すると、すべてのオブジェクト・グループに対してそのユーザーが登録されます。表内に無効なオブジェクト・グループがあると、表全体の登録に失敗します。  table_of_gnames に値を指定した場合は、list_of_gnames の値は指定しないでください。



## 例外

**表 35-7 REGISTER\_USER\_REPGROUP プロシージャの例外**

例外	説明
nonmaster	指定したオブジェクト・グループが存在していないか、または起動データベースがマスターではありません。
ORA-01917	ユーザーが存在しません。
typefailure	権限タイプの指定に誤りがあります。

## REVOKE\_ADMIN\_ANY\_SCHEMA プロシージャ

このプロシージャは、GRANT\_ADMIN\_ANY\_SCHEMA によって付与された権限とロールを、レプリケーション管理者から取り消します。

---

**注意：** GRANT\_ADMIN\_ANY\_SCHEMA とは別に付与された同一の権限とロールも取り消されます。

---

## 構文

```
DBMS_REPCAT_ADMIN.REVOKE_ADMIN_ANY_SCHEMA (  
    username IN VARCHAR2);
```

## パラメータ

**表 35-8 REVOKE\_ADMIN\_ANY\_SCHEMA プロシージャのパラメータ**

パラメータ	説明
username	権限を取り消すレプリケーション管理者の名前。

## 例外

**表 35-9 REVOKE\_ADMIN\_ANY\_SCHEMA プロシージャの例外**

例外	説明
ORA-01917	ユーザーが存在しません。

## REVOKE\_ADMIN\_SCHEMA プロシージャ

このプロシージャは、GRANT\_ADMIN\_SCHEMA によって付与された権限とロールを、レプリケーション管理者から取り消します。

**注意：** GRANT\_ADMIN\_SCHEMA とは別に付与された同一の権限とロールも取り消されます。

### 構文

```
DBMS_REPCAT_ADMIN.REVOKE_ADMIN_SCHEMA (  
    username IN VARCHAR2);
```

### パラメータ

表 35-10 REVOKE\_ADMIN\_SCHEMA プロシージャのパラメータ

パラメータ	説明
username	権限を取り消すレプリケーション管理者の名前。

### 例外

表 35-11 REVOKE\_ADMIN\_SCHEMA プロシージャの例外

例外	説明
ORA-01917	ユーザーが存在しません。

## UNREGISTER\_USER\_REPGROUP プロシージャ

このプロシージャは、REGISTER\_USER\_REPGROUP プロシージャによって付与された権限とロールを、代理スナップショット管理者または受信者から取り消します。

### 構文

```
DBMS_REPCAT_ADMIN.UNREGISTER_USER_REPGROUP (  
    username          IN   VARCHAR2,  
    privilege_type    IN   VARCHAR2,  
    list_of_gnames    IN   VARCHAR2 |  
    table_of_gnames   IN   dbms_utility.name_array);
```

## パラメータ

**表 35-12 UNREGISTER\_USER\_REPGROUP プロシージャのパラメータ**

パラメータ	説明
username	登録解除するユーザーの名前。
privilege_type	取り消す権限タイプ。privilege_type の定義には次の値を使用します。  RECEIVER: 受信者権限  PROXY_SNAPADMIN: 代理スナップショット管理者権限
list_of_gnames	ユーザーの受信者権限の登録を解除するオブジェクト・グループのカンマで区切られたリスト。リストのエントリ間に空白を挿入しないでください。NULL を設定すると、登録されているすべてのオブジェクト・グループに対する登録が解除されます。NULL を設定するには、名前表記法を使用する必要があります。リスト内に無効なオブジェクト・グループがあると、リスト全体の登録解除に失敗します。  list_of_gnames に値を指定した場合は、table_of_gnames の値は指定しないでください。
table_of_gnames	ユーザーの受信者権限の登録を解除するオブジェクト・グループの PL/SQL 表。PL/SQL 表の型は DBMS_UTILITY.NAME_ARRAY にしてください。この表は 1 が基準です。値に NULL を使用すると、登録されているすべてのオブジェクト・グループに対してそのユーザーの登録が解除されます。表内に無効なオブジェクト・グループがあると、表全体の登録解除に失敗します。  table_of_gnames に値を指定した場合は、list_of_gnames の値は指定しないでください。

## 例外

**表 35-13 UNREGISTER\_USER\_REPGROUP プロシージャの例外**

例外	説明
nonmaster	指定したオブジェクト・グループが存在していないか、または起動データベースがマスターではありません。
ORA-01917	ユーザーが存在しません。
typefailure	権限タイプの指定に誤りがあります。



---

## DBMS\_REPCAT\_INSTANTIATE

DBMS\_REPCAT\_INSTANTIATE パッケージは、配置テンプレートをインスタンス化します。

# サブプログラムの要約

表 36-1 DBMS\_REPCAT\_INSTANTIATE パッケージのサブプログラム

サブプログラム	説明
<a href="#">DROP_SITE_INSTANTIATION プロシージャ</a> (36-2 ページ)	DBA_REPCAT_TEMPLATE_SITES ビューからターゲット・サイトを削除するパブリック・プロシージャ。
<a href="#">INSTANTIATE_OFFLINE ファンクション</a> (36-3 ページ)	マスター・サイトでスクリプトを生成するパブリック・ファンクションで、オフライン時にリモート・スナップショット・サイトでスナップショット環境を作成するために使用します。
<a href="#">INSTANTIATE_ONLINE ファンクション</a> (36-5 ページ)	マスター・サイトでスクリプトを生成するパブリック・ファンクションで、オンライン時にリモート・スナップショット・サイトでスナップショット環境を作成するために使用します。

## DROP\_SITE\_INSTANTIATION プロシージャ

このプロシージャは、ターゲット・サイトのテンプレート・インスタンスエーションを削除します。マスター・サイトの関連メタデータをすべて削除し、指定したサイトにおけるスナップショットのリフレッシュを使用禁止にします。このプロシージャは、テンプレートを最初にインスタンス化したユーザーとして実行する必要があります。

**注意：** テンプレートをインスタンス化したユーザーを調べるには、REPCAT\_TEMPLATE\_SITES ビューを参照してください。

### 構文

```
DBMS_REPCAT_INSTANTIATE.DROP_SITE_INSTANTIATION (  
    refresh_template_name IN VARCHAR2,  
    site_name              IN VARCHAR2,  
    repapi_site_id         IN NUMBER := -1e-130);
```

### パラメータ

表 36-2 DROP\_SITE\_INSTANTIATION プロシージャのパラメータ

パラメータ	説明
refresh_template_name	削除する配置テンプレートの名前。
site_name	指定したテンプレート・インスタンスエーションを削除する Oracle Server のサイト。( site_name を指定した場合、 repapi_site_id は指定しないでください。)

表 36-2 DROP\_SITE\_INSTANTIATION プロシージャのパラメータ

パラメータ	説明
repapi_site_id	指定したテンプレート・インスタンスエーションを削除する REPAPI の位置を示します。(repapi_site_id を指定した場合、site_name は指定しないでください。)

## INstantiate\_Offline ファンクション

このファンクションは、マスター・サイトでスクリプトを生成します。このスクリプトは、オフライン時にリモート・スナップショット・サイトでスナップショット環境を作成するために使用します。生成されたスクリプトは、長時間マスター・サイトに接続したままにできないリモート・スナップショット・サイトで使用してください。これは、スナップショット・サイトがラップトップの場合に便利な方法です。Replication Manager のパッケージ作成ツールを使用して、生成したスクリプトとデータを 1 つのファイルにパッケージ化すると、このファイルを FTP サイトにポストしたり、CD-ROM やフロッピー・ディスクなどにロードすることができます。詳細は、『Oracle8i レプリケーション・ガイド』の「配置テンプレート」を参照してください。

このファンクションで生成されたスクリプトは、USER\_REPCAT\_TEMP\_OUTPUT 一時ビューに格納され、配置テンプレートの配布時に、Replication Manager などの Oracle Tools で使用されます。このファンクションで戻される数値は、USER\_REPCAT\_TEMP\_OUTPUT ビューから該当する情報を取り出すために使用されます。

このパブリック・ファンクションを実行するユーザーは、指定したサイトでインスタンス化されたテンプレートの登録ユーザーになります。

---

**注意：** このファンクションは、配置テンプレートのオフライン・インスタンスエーションの実行に使用されます。

---

このファンクションを、[DBMS\\_OFFLINE\\_OG](#) パッケージ内のプロシージャ（マスター表のオフライン・インスタンスエーションの実行に使用）や [DBMS\\_OFFLINE\\_SNAPSHOT](#) パッケージ内のプロシージャ（スナップショットのオフライン・インスタンスエーションの実行に使用）と混同しないでください。これらの使用方法の詳細は、各パッケージの説明を参照してください。

---

構文

```
DBMS_REPCAT_INSTANTIATE.INSTANTIATE_OFFLINE(  
    refresh_template_name  IN   VARCHAR2,  
    site_name              IN   VARCHAR2,  
    runtime_parm_id        IN   NUMBER      := -1e-130,  
    next_date              IN   DATE        := SYSDATE,  
    interval               IN   VARCHAR2    := 'SYSDATE + 1')  
RETURN NUMBER;
```

パラメータ

表 36-3 INSTANTIATE\_OFFLINE ファンクションのパラメータ

パラメータ	説明
refresh_template_name	インスタンス化する配置テンプレートの名前。
site_name	配置テンプレートをインスタンス化するリモート・サイトの名前。
runtime_parm_id	INSERT_RUNTIME_PARMS プロシージャを使用してランタイム・パラメータ値を定義した場合は、ランタイム・パラメータ値の作成時に使用した ID を指定します。( GET_RUNTIME_PARM_ID ファンクションを使用して取り出された ID です。)
next_date	リフレッシュ・グループの作成時に使用される次回リフレッシュ日付の値を指定します。
interval	リフレッシュ・グループの作成時に使用されるリフレッシュ間隔を指定します。

例外

表 36-4 INSTANTIATE\_OFFLINE ファンクションの例外

例外	説明
miss_refresh_template	指定した配置テンプレート名が無効か、または存在しません。
miss_user	認可ユーザー名が無効か、または存在しません。指定したユーザーが DBA_REPCAT_TEMPLATE_AUTH ビューにリストされていることを検証してください。リストされていない場合、指定したユーザーは、対象とする配置テンプレートのインスタンス化を認可されていません。
bad_parms	定義したユーザー・パラメータ値またはテンプレート・デフォルト値によって移入されていないテンプレート・パラメータがあります。事前定義値の数とテンプレート・パラメータ数が一致していないか、または事前定義値がターゲット・パラメータに対して無効です(型の不一致など)。



戻り値

表 36-5 INSTANTIATE\_OFFLINE ファンクションの戻り値

戻り値	説明
< システム生成番号 >	システムによって生成された番号。生成されたインスタンスエーショ ン・スクリプトを取り出すため USER_REPCAT_TEMP_OUTPUT ビューから選択するときに、output_id に使用する値です。

INSTANTIATE\_ONLINE ファンクション

このファンクションは、マスター・サイトでスクリプトを生成します。このスクリプトは、オンライン時にリモート・スナップショット・サイトでスナップショット環境を作成するために使用します。リモート・スナップショット・サイトでインスタンス化プロセスは長くなる場合があるため（必要な時間は新しいスナップショットに移入されるデータの量によって異なります）生成されたスクリプトは、マスター・サイトに長時間接続したままにできるリモート・スナップショット・サイトで使用してください。

このファンクションで生成されたスクリプトは、USER\_REPCAT\_TEMP\_OUTPUT 一時ビューに格納され、配置テンプレートの配布時に、Replication Manager などの Oracle Tools で使用されます。このファンクションで戻される数値は、USER\_REPCAT\_TEMP\_OUTPUT ビューから該当する情報を取り出すために使用されます。

このパブリック・ファンクションを実行するユーザーは、指定したサイトでインスタンス化されたテンプレートの登録ユーザーになります。

構文

```
DBMS_REPCAT_INSTANTIATE.INSTANTIATE_ONLINE(  
    refresh_template_name    IN    VARCHAR2,  
    site_name                IN    VARCHAR2,  
    runtime_parm_id          IN    NUMBER    := -1e-130,  
    next_date                IN    DATE      := SYSDATE,  
    interval                  IN    VARCHAR2 := 'SYSDATE + 1')  
RETURN NUMBER;
```

パラメータ

表 36-6 INSTANTIATE\_ONLINE ファンクションのパラメータ

パラメータ	説明
refresh_template_name	インスタンス化する配置テンプレートの名前。
site_name	配置テンプレートをインスタンス化するリモート・サイトの名前。
runtime_parm_id	INSERT_RUNTIME_PARMS プロシーダを使用してランタイム・パラメータ値を定義した場合は、ランタイム・パラメータの作成時に使用した ID ( GET_RUNTIME_PARM_ID ファンクションを使用して取り出された ID です ) を指定します。
next_date	リフレッシュ・グループの作成時に使用される次回リフレッシュ日付の値を指定します。
interval	リフレッシュ・グループの作成時に使用されるリフレッシュ間隔を指定します。

例外

表 36-7 INSTANTIATE\_ONLINE ファンクションの例外

例外	説明
miss_refresh_template	指定した配置テンプレート名が無効か、または存在しません。
miss_user	認可ユーザー名が無効か、または存在しません。指定したユーザーが DBA_REPCAT_TEMPLATE_AUTH ビューにリストされていることを検証してください。リストされていない場合、指定したユーザーは、ターゲット配置テンプレートのインスタンス化を認可されていません。
bad_parms	定義したユーザー・パラメータ値またはテンプレート・デフォルト値によって移入されていないテンプレート・パラメータがあります。事前定義値の数とテンプレート・パラメータ数が一致していないか、または事前定義値がターゲット・パラメータに対して無効です ( 型の不一致など )。

戻り値

表 36-8 INSTANTIATE\_ONLINE ファンクションの戻り値

戻り値	説明
< システム生成番号 >	システムによって生成された番号。生成されたインスタンスエーション・スクリプトを取り出すため USER_REPCAT_TEMP_OUTPUT ビューから選択するときに、output_id に使用する値です。

---

## DBMS\_REPCAT\_RGT

DBMS\_REPCAT\_RGT は、リフレッシュ・グループ・テンプレートのメンテナンスと定義を制御します。

# サブプログラムの要約

表 37-1 DBMS\_REPCAT\_RGT パッケージのサブプログラム

サブプログラム	説明
<a href="#">ALTER_REFRESH_TEMPLATE プロシージャ</a> (37-4 ページ)	DBA が既存の配置テンプレートを変更できます。
<a href="#">ALTER_TEMPLATE_OBJECT プロシージャ</a> (37-6 ページ)	指定した配置テンプレートに追加されているオブジェクトを変更します。
<a href="#">ALTER_TEMPLATE_PARM プロシージャ</a> (37-9 ページ)	DBA が特定の配置テンプレートのパラメータを変更できます。
<a href="#">ALTER_USER_AUTHORIZATION プロシージャ</a> (37-10 ページ)	DBA_REPCAT_TEMPLATE_AUTH ビューの内容を変更します。
<a href="#">ALTER_USER_PARM_VALUE プロシージャ</a> (37-11 ページ)	特定のユーザーに対して定義されている既存のパラメータ値を変更します。
<a href="#">COMPARE_TEMPLATES ファンクション</a> (37-14 ページ)	DBA が 2 つの配置テンプレートの内容を比較できます。
<a href="#">COPY_TEMPLATE ファンクション</a> (37-15 ページ)	DBA が配置テンプレートをコピーできます。
<a href="#">CREATE_OBJECT_FROM_EXISTING ファンクション</a> (37-17 ページ)	既存のデータベース・オブジェクトからテンプレート・オブジェクト定義を作成し、それをターゲット配置テンプレートに追加します。
<a href="#">CREATE_REFRESH_TEMPLATE ファンクション</a> (37-18 ページ)	配置テンプレートを作成します。テンプレート名、プライベートまたはパブリック・ステータス、およびターゲット・リフレッシュ・グループを定義できます。
<a href="#">CREATE_TEMPLATE_OBJECT ファンクション</a> (37-20 ページ)	ターゲット配置テンプレートのコンテナにオブジェクト定義を追加します。
<a href="#">CREATE_TEMPLATE_PARM ファンクション</a> (37-22 ページ)	特定の配置テンプレートのパラメータを作成します。この結果、リモート・スナップショット・サイトでカスタム・データ・セットを作成できます。
<a href="#">CREATE_USER_AUTHORIZATION ファンクション</a> (37-24 ページ)	特定のユーザーに、プライベート配置テンプレートのインスタンス化を認可します。
<a href="#">CREATE_USER_PARM_VALUE ファンクション</a> (37-26 ページ)	特定のユーザーに対する配置テンプレートのパラメータ値を事前定義します。
<a href="#">DELETE_RUNTIME_PARMS プロシージャ</a> (37-28 ページ)	INSERT_RUNTIME_PARMS プロシージャを使用して定義したランタイム・パラメータ値を削除します。
<a href="#">DROP_ALL_OBJECTS プロシージャ</a> (37-28 ページ)	DBA が、配置テンプレートからすべてのオブジェクトまたは特定のオブジェクト型を削除できます。

表 37-1 DBMS\_REPCAT\_RGT パッケージのサブプログラム

サブプログラム	説明
<a href="#">DROP_ALL_TEMPLATE_PARS プロシージャ</a> (37-29 ページ)	DBA が、指定した配置テンプレートのテンプレート・パラメータを削除できます。
<a href="#">DROP_ALL_TEMPLATE_SITES プロシージャ</a> (37-30 ページ)	DBA_REPCAT_TEMPLATE_SITES ビューからすべてのエントリを削除します。
<a href="#">DROP_ALL_TEMPLATES プロシージャ</a> (37-31 ページ)	プロシージャがコールされるサイトの配置テンプレートをすべて削除します。
<a href="#">DROP_ALL_USER_AUTHORIZATIONS プロシージャ</a> (37-31 ページ)	DBA が、指定した配置テンプレートに対するユーザー認証をすべて削除できます。
<a href="#">DROP_ALL_USER_PARM_VALUES プロシージャ</a> (37-32 ページ)	特定の配置テンプレートに対するユーザー・パラメータ値を削除します。
<a href="#">DROP_REFRESH_TEMPLATE プロシージャ</a> (37-33 ページ)	配置テンプレートを削除します。
<a href="#">DROP_SITE_INSTANTIATION プロシージャ</a> (37-34 ページ)	DBA_REPCAT_TEMPLATE_SITES ビューからターゲット・サイトを削除します。
<a href="#">DROP_TEMPLATE_OBJECT プロシージャ</a> (37-35 ページ)	特定の配置テンプレートからテンプレート・オブジェクトを削除します。
<a href="#">DROP_TEMPLATE_PARM プロシージャ</a> (37-36 ページ)	DBA_REPCAT_TEMPLATE_PARS ビューから既存のテンプレート・パラメータを削除します。
<a href="#">DROP_USER_AUTHORIZATION プロシージャ</a> (37-36 ページ)	DBA_REPCAT_TEMPLATE_AUTH ビューからユーザー認証エントリを削除します。
<a href="#">DROP_USER_PARM_VALUE プロシージャ</a> (37-37 ページ)	特定の配置テンプレートに対する事前定義ユーザー・パラメータ値を削除します。
<a href="#">GET_RUNTIME_PARM_ID ファンクション</a> (37-38 ページ)	ランタイム・パラメータ値の定義時に使用する ID を取り出します。
<a href="#">INSERT_RUNTIME_PARS プロシージャ</a> (37-39 ページ)	テンプレートのインスタンス化前にランタイム・パラメータ値を定義します。
<a href="#">INstantiate_OFFLINE ファンクション</a> (37-40 ページ)	マスター・サイトでスクリプトを生成します。このスクリプトは、オフライン時にリモート・スナップショット・サイトでスナップショット環境を作成するために使用します。
<a href="#">INstantiate_ONLINE ファンクション</a> (37-42 ページ)	マスター・サイトでスクリプトを生成します。このスクリプトは、オンライン時にリモート・スナップショット・サイトでスナップショット環境を作成するために使用します。

表 37-1 DBMS\_REPCAT\_RGT パッケージのサブプログラム

サブプログラム	説明
<a href="#">LOCK_TEMPLATE_EXCLUSIVE プロシージャ</a> (37-44 ページ)	配置テンプレートの更新中または変更中に、ユーザーがテンプレートの読み込みまたはインスタンス化を実行できないようにします。
<a href="#">LOCK_TEMPLATE_SHARED プロシージャ</a> (37-45 ページ)	指定した配置テンプレートを読み取り専用にします。

ALTER\_REFRESH\_TEMPLATE プロシージャ

このプロシージャでは、DBA が既存の配置テンプレートを変更できます。新規配置テンプレート名、新規リフレッシュ・グループまたは新規所有者の定義や、パブリックまたはプライベート・ステータスの変更などを実行できます。

構文

```
DBMS_REPCAT_RGT.ALTER_REFRESH_TEMPLATE (  
    refresh_template_name      IN    VARCHAR2,  
    new_owner                  IN    VARCHAR2 := '-',  
    new_refresh_group_name     IN    VARCHAR2 := '-',  
    new_refresh_template_name  IN    VARCHAR2 := '-',  
    new_template_comment      IN    VARCHAR2 := '-',  
    new_public_template        IN    VARCHAR2 := '-',  
    new_last_modified          IN    DATE      := to_date('1', 'J'),  
    new_modified_by            IN    NUMBER    := -1e-130);
```

## パラメータ

表 37-2 ALTER\_REFRESH\_TEMPLATE プロシージャのパラメータ

パラメータ	説明
refresh_template_name	変更する配置テンプレートの名前。
new_owner	配置テンプレートの新しい所有者名。現行所有者を変更しないときは値を指定しないでください。
new_refresh_group_name	必要な場合は、テンプレート・オブジェクトが追加される新規リフレッシュ・グループ名を指定します。現行のリフレッシュ・グループを変更しないときは値を指定しないでください。
new_refresh_template_name	新しい配置テンプレート名を指定します。現行の配置テンプレート名を変更しないときは値を指定しないでください。
new_template_comment	配置テンプレートの新しいコメント。現行のテンプレート・コメントを変更しないときは値を指定しないでください。
new_public_template	配置テンプレートがパブリックかプライベートかを決定します。指定できる値は 'Y' と 'N' ('Y' = パブリック、'N' = プライベート) のみです。現行の値を変更しないときは値を指定しないでください。
new_last_modified	この配置テンプレートの最終更新日付。値を指定しないと、現行の日付が自動的に使用されます。
new_modified_by	この配置テンプレートの最終更新ユーザーの名前。値を指定しないと、現行ユーザーが自動的に使用されます。

## 例外

表 37-3 ALTER\_REFRESH\_TEMPLATE プロシージャの例外

例外	説明
miss_refresh_template	指定した配置テンプレート名が無効か、または存在しません。
bad_public_template	public_template パラメータの指定に誤りがあります。public_template パラメータには、'Y' (パブリック・テンプレート) または 'N' (プライベート・テンプレート) のいずれかを指定してください。
dupl_refresh_template	指定した名前のテンプレートがすでに存在します。DBA_REPCAT_REFRESH_TEMPLATES ビューを参照してください。

## ALTER\_TEMPLATE\_OBJECT プロシージャ

このプロシージャは、指定した配置テンプレートに追加されているオブジェクトを変更します。オブジェクト DDL の変更や、異なる配置テンプレートへのオブジェクトの割当てを実行できます。

テンプレートの変更内容は、変更後に配置テンプレートをインスタンス化する新規サイトでのみ反映されます。テンプレートをすでにインスタンス化しているリモート・サイトは、配置テンプレートを再インスタンス化して変更内容を適用する必要があります。

### 構文

```
DBMS_REPCAT_RGT.ALTER_TEMPLATE_OBJECT (  
    refresh_template_name      IN   VARCHAR2,  
    object_name                 IN   VARCHAR2,  
    object_type                 IN   VARCHAR2,  
    new_refresh_template_name   IN   VARCHAR2 := '-',  
    new_object_name             IN   VARCHAR2 := '-',  
    new_object_type             IN   VARCHAR2 := '-',  
    new_ddl_text                IN   CLOB      := '-',  
    new_master_rollback_seg     IN   VARCHAR2 := '-',  
    new_flavor_id               IN   NUMBER    := -1e-130);
```



パラメータ

表 37-4 ALTER\_TEMPLATE\_OBJECT プロシージャのパラメータ

パラメータ	説明
refresh_template_name	変更するオブジェクトを含んだ配置テンプレートの名前。
object_name	変更するテンプレート・オブジェクトの名前。
object_type	変更するオブジェクトの型。
new_refresh_template_name	このオブジェクトを再割当てする新しい配置テンプレートの名前。オブジェクトを現行の配置テンプレートに割り当てたままにするときは値を指定しないでください。
new_object_name	テンプレート・オブジェクトの新しい名前。現行のオブジェクト名を変更しないときは値を指定しないでください。
new_object_type	必要な場合は新しいオブジェクト型を指定します。次の型のオブジェクトを指定できます。 <div><div>SNAPSHOT</div><div>INDEX</div><div>TABLE</div><div>VIEW</div><div>SYNONYM</div><div>SEQUENCE</div><div>PROCEDURE</div><div>FUNCTION</div><div>PACKAGE</div><div>PACKAGE BODY</div><div>TRIGGER</div><div>DATABASE LINK</div></div>
new_ddl_text	指定したオブジェクトの新しいオブジェクト DDL。現行のオブジェクト DDL を変更しないときは値を指定しないでください。
new_master_rollback_seg	指定したオブジェクトの新しいマスター・ロールバック・セグメント。現行のロールバック・セグメントを変更しないときは値を指定しないでください。
new_flavor_id	指定したオブジェクトの新しい固有 ID。現行の固有 ID を変更しないときは値を指定しないでください。

例外

表 37-5 ALTER\_TEMPLATE\_OBJECT プロシージャの例外

例外	説明
miss_refresh_template	指定した配置テンプレート名が無効か、または存在しません。
miss_flavor_id	指定した固有 ID が無効か、または存在しません。
bad_object_type	オブジェクト型の指定に誤りがあります。有効なオブジェクト型のリストは、表 37-4 を参照してください。
miss_template_object	指定したテンプレート・オブジェクト名が無効か、または存在しません。
dupl_template_object	new_refresh_template_name パラメータに指定した新規テンプレート名はすでに存在しています。

使用上の注意

ALTER\_TEMPLATE\_OBJECT プロシージャは CLOB を利用するため、このプロシージャを使用する場合は、DBMS\_LOB パッケージを利用する必要があります。次の例は、ALTER\_TEMPLATE\_OBJECT プロシージャで DBMS\_LOB パッケージを使用する方法を示しています。

```
DECLARE
    tempstring VARCHAR2(100);
    templob CLOB;
BEGIN
    DBMS_LOB.CREATETEMPORARY(templob, TRUE, DBMS_LOB.SESSION);
    tempstring := 'CREATE SNAPSHOT snap_sales AS SELECT *
        FROM sales WHERE salesperson = :salesid and region_id = :region';
    DBMS_LOB.WRITE(templob, length(tempstring), 1, tempstring);
    DBMS_REPCAT_RGT.ALTER_TEMPLATE_OBJECT(
        refresh_template_name => 'rgt_personnel',
        object_name => 'SNAP_SALES',
        object_type => 'SNAPSHOT',
        new_ddl_text => templob);
    DBMS_LOB.FREETEMPORARY(templob);
END;
/
```

## ALTER\_TEMPLATE\_PARM プロシージャ

このプロシージャでは、DBA が特定の配置テンプレートのパラメータを変更できます。パラメータ名の変更またはデフォルト値やプロンプト文字列の再定義などを実行できます。

### 構文

```
DBMS_REPCAT_RGT.ALTER_TEMPLATE_PARM (  
    refresh_template_name      IN   VARCHAR2,  
    parameter_name             IN   VARCHAR2,  
    new_refresh_template_name  IN   VARCHAR2 := '-',  
    new_parameter_name         IN   VARCHAR2 := '-',  
    new_default_parm_value     IN   CLOB      := NULL,  
    new_prompt_string          IN   VARCHAR2 := '-',  
    new_user_override          IN   VARCHAR2 := '-');
```

### パラメータ

表 37-6 ALTER\_TEMPLATE\_PARM プロシージャのパラメータ

パラメータ	説明
refresh_template_name	変更するパラメータを含んだ配置テンプレートの名前。
parameter_name	変更するパラメータの名前。
new_refresh_template_name	指定したパラメータを再割当てする配置テンプレートの名前（あるテンプレートのパラメータを別のテンプレートに移動するときに便利です）。パラメータを現行テンプレートに割り当てたままにするときは値を指定しないでください。
new_parameter_name	テンプレート・パラメータの新しい名前。現行のパラメータ名を変更しないときは値を指定しないでください。
new_default_parm_value	指定したパラメータの新しいデフォルト値。現行のデフォルト値を変更しないときは値を指定しないでください。
new_prompt_string	指定したパラメータの新しいプロンプト・テキスト。現行のプロンプト文字列を変更しないときは値を指定しないでください。
new_user_override	インスタンス化プロセス時にプロンプトが表示された場合に、ユーザーにデフォルト値の上書きを許可するかどうかを決定します（該当のパラメータに対してユーザー・パラメータ値が定義されていないと、プロンプトが表示されます）。デフォルト値の上書きを許可する場合は 'Y'、許可しない場合は 'N' を設定します。

例外

表 37-7 ALTER\_TEMPLATE\_PARM プロシージャの例外

例外	説明
miss_refresh_template	指定した配置テンプレート名が無効か、または存在しません。
miss_template_parm	指定したテンプレート・パラメータが無効か、または存在しません。
dupl_template_parm	new_refresh_template_name と new_parameter_name の組合せがすでに存在します。

使用上の注意

ALTER\_TEMPLATE\_PARM プロシージャは CLOB を利用するため、このプロシージャを使用する場合は、DBMS\_LOB パッケージを利用する必要があります。次の例は、ALTER\_TEMPLATE\_PARM プロシージャで DBMS\_LOB パッケージを使用する方法を示しています。

```
DECLARE
    tempstring VARCHAR2(100);
    templob CLOB;
BEGIN
    DBMS_LOB.CREATETEMPORARY(templob, TRUE, DBMS_LOB.SESSION);
    tempstring := 'REGION 20';
    DBMS_LOB.WRITE(templob, length(tempstring), 1, tempstring);
    DBMS_REPCAT_RGT.ALTER_TEMPLATE_PARM(
        refresh_template_name => 'rgt_personnel',
        parameter_name => 'region',
        new_default_parm_value => templob);
    DBMS_LOB.FREETEMPORARY(templob);
END;
```

ALTER\_USER\_AUTHORIZATION プロシージャ

このプロシージャは、DBA\_REPCAT\_TEMPLATE\_AUTH ビューの内容を変更します。具体的には、ユーザー・テンプレートまたは配置テンプレートの認可割当てを変更できます。たとえば、このプロシージャは、従業員が異動して、別の配置テンプレートのスナップショット環境を必要とする場合に役立ちます。DBA が新規配置テンプレートを従業員に割り当てるだけで、そのユーザーはターゲット・テンプレートのインスタンス化を認可されます。

## 構文

```
DBMS_REPCAT_RGT.ALTER_USER_AUTHORIZATION (
    user_name            IN    VARCHAR2,
    refresh_template_name IN    VARCHAR2,
    new_user_name        IN    VARCHAR2 := '-',
    new_refresh_template_name IN    VARCHAR2 := '-');
```

## パラメータ

表 37-8 ALTER\_USER\_AUTHORIZATION プロシージャのパラメータ

パラメータ	説明
user_name	認可を変更対象とするユーザーの名前。
refresh_template_name	指定したユーザーに現在割り当てられている、変更対象の配置テンプレートの名前。
new_user_name	このパラメータは、このテンプレート認可に対して、新しいユーザーを定義するときに使用します。現行ユーザーを変更しないときは値を指定しないでください。
new_refresh_template_name	指定したユーザー（既存のユーザー、または指定した新規ユーザー）がインスタンス化を認可される配置テンプレート。現行の配置テンプレートを変更しないときは値を指定しないでください。

## 例外

表 37-9 ALTER\_USER\_AUTHORIZATION プロシージャの例外

例外	説明
miss_user_authorization	指定した user_name と refresh_template_name の組合せが DBA_REPCAT_TEMPLATE_AUTH ビューに存在しません。
miss_user	new_user_name または user_name パラメータに指定したユーザー名が無効か、または存在しません。
miss_refresh_template	new_refresh_template パラメータに指定した配置テンプレートが無効か、または存在しません。
dupl_user_authorization	指定したユーザー名と配置テンプレート名に対する行がすでに存在しています。DBA_REPCAT_AUTH_TEMPLATES ビューを参照してください。

## ALTER\_USER\_PARM\_VALUE プロシージャ

このプロシージャは、特定のユーザーに対して定義されている既存のパラメータ値を変更します。スナップショット環境で割当て表を使用している場合に特に便利なプロシージャで

す。ユーザー・パラメータ値を変更するだけで、リモート・スナップショット・サイトのデータ・セットを安全にすばやく変更できます。

割当て表の使用方法的詳細は、『Oracle8i レプリケーション・ガイド』の「配置テンプレートの設計」を参照してください。

構文

```
DBMS_REPCAT_RGT.ALTER_USER_PARM_VALUE(  
  refresh_template_name      IN   VARCHAR2,  
  parameter_name             IN   VARCHAR2,  
  user_name                  IN   VARCHAR2,  
  new_refresh_template_name  IN   VARCHAR2 := '-',  
  new_parameter_name         IN   VARCHAR2 := '-',  
  new_user_name              IN   VARCHAR2 := '-',  
  new_parm_value             IN   CLOB      := NULL);
```

パラメータ

表 37-10 ALTER\_USER\_PARM\_VALUE プロシージャのパラメータ

パラメータ	説明
refresh_template_name	変更するユーザー・パラメータ値を含んだ配置テンプレートの名前。
parameter_name	変更するパラメータの名前。
user_name	パラメータ値を変更対象とするユーザーの名前。
new_refresh_template_name	指定したユーザー・パラメータ値を再割当てする配置テンプレートの名前（別のテンプレートに対してユーザーを認可するときに便利です）。パラメータを現行テンプレートに割り当てたままにするときは値を指定しないでください。
new_parameter_name	新規テンプレート・パラメータ名。既存のパラメータに対して定義されたユーザー値を変更しないときは値を指定しないでください。
new_user_name	このパラメータ値を適用する新規ユーザー名。パラメータを現行ユーザーに割り当てたままにするときは値を指定しないでください。
new_parm_value	指定したユーザー・パラメータの新しいパラメータ値。現行のパラメータ値を変更しないときは値を指定しないでください。

例外

表 37-11 ALTER\_USER\_PARM\_VALUE プロシージャの例外

例外	説明
miss_refresh_template	指定した配置テンプレート名が無効か、または存在しません。
miss_template_parm	指定したテンプレート・パラメータが無効か、または存在しません。
miss_user	user_name または new_user_name パラメータに指定したユーザー名が無効か、または存在しません。
miss_user_parm_values	指定したユーザー・パラメータ値が存在しません。
dupl_user_parm_values	指定した新規ユーザー・パラメータはすでに存在しています。

使用上の注意

ALTER\_USER\_PARM\_VALUE プロシージャは CLOB を利用するため、このプロシージャを使用する場合は、DBMS\_LOB パッケージを利用する必要があります。次の例は、ALTER\_USER\_PARM\_VALUE プロシージャで DBMS\_LOB パッケージを使用する方法を示しています。

```
DECLARE
    tempstring VARCHAR2(100);
    templob CLOB;
BEGIN
    DBMS_LOB.CREATETEMPORARY(templob, TRUE, DBMS_LOB.SESSION);
    tempstring := 'REGION 20';
    DBMS_LOB.WRITE(templob, length(tempstring), 1, tempstring);
    DBMS_REPCAT_RGT.ALTER_USER_PARM_VALUE(
        refresh_template_name => 'rgt_personnel',
        parameter_name => 'region',
        user_name => 'BOB',
        new_parm_value => templob);
    DBMS_LOB.FREETEMPORARY(templob);
END;
/
```

## COMPARE\_TEMPLATES ファンクション

このファンクションでは、DBA が 2 つの配置テンプレートの内容を比較できます。2 つの配置テンプレートの相違点は、USER\_REPCAT\_TEMP\_OUTPUT 表に格納されます。

COMPARE\_TEMPLATES ファンクションは数値を戻します。この数値は、USER\_REPCAT\_TEMP\_OUTPUT 表の問合せ時に WHERE 句に指定する値です。たとえば、COMPARE\_TEMPLATES プロシージャが 10 を戻した場合、指定した 2 つのテンプレート間の相違点をすべて表示するには、次の SELECT 文を実行します（テンプレートが同じ場合、SELECT 文で行は戻されません）。

```
SELECT text FROM USER_REPCAT_TEMP_OUTPUT
WHERE output_id = 10 ORDER BY LINE;
```

USER\_REPCAT\_TEMP\_OUTPUT の内容は、ユーザーが切断するか、または ROLLBACK が実行されると失われます。

### 構文

```
DBMS_REPCAT_RGT.COMPARE_TEMPLATES (
    source_template_name    IN    VARCHAR2,
    compare_template_name   IN    VARCHAR2)
RETURN NUMBER;
```

### パラメータ

表 37-12 COMPARE\_TEMPLATES ファンクションのパラメータ

パラメータ	説明
source_template_name	比較対象の最初の配置テンプレート名。
compare_template_name	比較対象の 2 番目の配置テンプレート名。

### 例外

表 37-13 COMPARE\_TEMPLATES ファンクションの例外

例外	説明
miss_refresh_template	比較対象の配置テンプレート名が無効か、または存在しません。



戻り値

表 37-14 COMPARE\_TEMPLATES ファンクションの戻り値

戻り値	説明
< システム生成番号 >	システムによって生成された数値。比較したテンプレート間の相違点を参照するため USER_REPCAT_TEMP_OUTPUT ビューから選択するときに、output_id に使用する値です。

COPY\_TEMPLATE ファンクション

このファンクションでは、DBA が配置テンプレートをコピーできます。このファンクションは、新しく作成する配置テンプレートに、既存の配置テンプレートに含まれているオブジェクトを多数使用する場合に便利です。このファンクションは、配置テンプレート、テンプレート・オブジェクト、テンプレート・パラメータおよびユーザー・パラメータ値をコピーします。DBA は、オプションでこのテンプレートに対するユーザー認証をコピーできます。このファンクションで戻される番号は、配置テンプレートを管理するために Oracle で内部的に使用されます。

**注意：** DBA\_REPCAT\_TEMPLATE\_SITES ビュー内の値はコピーされません。

このファンクションでは、配置テンプレートを別のマスター・サイトにコピーすることもできます。この機能は、配置テンプレートの配布、および複数サイト間でのネットワーク負荷の分散に役立ちます。

構文

```
DBMS_REPCAT_RGT.COPY_TEMPLATE (
  old_refresh_template_name    IN   VARCHAR2 ,
  new_refresh_template_name    IN   VARCHAR2 ,
  copy_user_authorizations     IN   VARCHAR2 ,
  dblink                       IN   VARCHAR2 := NULL)
RETURN NUMBER;
```

パラメータ

表 37-15 COPY\_TEMPLATE ファンクションのパラメータ

パラメータ	説明
old_refresh_template_name	コピー元の配置テンプレート名。
new_refresh_template_name	新規配置テンプレート名。
copy_user_authorizations	コピー元のテンプレートのテンプレート認可を新しい配置テンプレート用にコピーするかどうかを指定します。有効な値は、'Y'、'N' および NULL です。  <b>注意：</b> すべてのユーザーがターゲット・データベースに存在している必要があります。
dblink	オプションで、配置テンプレートのコピー元を定義します（この機能は、他のマスター・サイトに配置テンプレートを配布するときに便利です）。指定しないと、配置テンプレートはローカル・マスター・サイトからコピーされます。

例外

表 37-16 COPY\_TEMPLATE ファンクションの例外

例外	説明
miss_refresh_template	コピー元の配置テンプレート名が無効か、または存在しません。
dupl_refresh_template	指定した新規リフレッシュ・テンプレート名がすでに存在していません。
bad_copy_auth	copy_user_authorization パラメータに指定した値が無効です。有効な値は、'Y'、'N' および NULL です。

戻り値

表 37-17 COPY\_TEMPLATES ファンクションの戻り値

戻り値	説明
< システム生成番号 >	システム生成番号は、Oracle で内部的に使用されます。

## CREATE\_OBJECT\_FROM\_EXISTING ファンクション

このファンクションは、既存のデータベース・オブジェクトからテンプレート・オブジェクト定義を作成し、それをターゲット配置テンプレートに追加します。ターゲット配置テンプレートがリモート・スナップショット・サイトでインスタンス化されるときに、元のデータベース・オブジェクトを作成したオブジェクト DDL が実行されます。これは、既存のトリガーやプロシージャをテンプレートに追加するときに便利な方法です。このファンクションで戻される番号は、配置テンプレートを管理するために Oracle で内部的に使用されます。

### 構文

```
DBMS_REPCAT_RGT.CREATE_OBJECT_FROM_EXISTING(  
    refresh_template_name  IN   VARCHAR2,  
    object_name            IN   VARCHAR2,  
    sname                  IN   VARCHAR2,  
    oname                  IN   VARCHAR2,  
    otype                  IN   VARCHAR2)  
RETURN NUMBER;
```

### パラメータ

表 37-18 CREATE\_OBJECT\_FROM\_EXISTING ファンクションのパラメータ

パラメータ	説明
refresh_template_name	このオブジェクトを追加する配置テンプレートの名前。
object_name	必要の場合は、配置テンプレートに追加する既存オブジェクトの新規名を指定します（既存のオブジェクトに新しい名前を定義できます）。
sname	テンプレート・オブジェクトの作成元のオブジェクトを含んだスキーマ。
oname	テンプレート・オブジェクトの作成元のオブジェクト名。
otype	テンプレートに追加するデータベース・オブジェクトの型（PROCEDURE、TRIGGER など）。オブジェクト型は、次の数値型識別子を使用して指定する必要があります（DATABASE LINK または SNAPSHOT は、このファンクションでは有効なオブジェクト型ではありません）。
	SEQUENCE                      PROCEDURE
	INDEX                          FUNCTION
	TABLE                          PACKAGE
	VIEW                           PACKAGE BODY
	SYNONYM                       TRIGGER

例外

表 37-19 CREATE\_OBJECT\_FROM\_EXISTING ファンクションの例外

例外	説明
miss_refresh_template	指定したリフレッシュ・テンプレート名が無効か、または存在しません。既存の配置テンプレートのリストは、DBA_REPCAT_REFRESH_TEMPLATE ビューを参照してください。
bad_object_type	オブジェクト型の指定に誤りがあります（詳細は、表 37-24 を参照してください）。
dupl_template_object	同じ名前と型のオブジェクトが、指定した配置テンプレートにすでに追加されています。
objectmissing	指定した既存オブジェクトが存在しません。

戻り値

表 37-20 CREATE\_OBJECT\_FROM\_EXISTING ファンクションの戻り値

戻り値	説明
< システム生成番号 >	システム生成番号は、Oracle で内部的に使用されます。

CREATE\_REFRESH\_TEMPLATE ファンクション

このファンクションは、配置テンプレートを作成します。テンプレート名、プライベートまたはパブリック・ステータス、およびターゲット・リフレッシュ・グループを定義できます。テンプレート・オブジェクト、ユーザー認証またはテンプレート・パラメータを作成するたびに、このファンクションで作成された配置テンプレートを参照します。このファンクションは、DBA\_REPCAT\_REFRESH\_TEMPLATES ビューに行を追加します。このファンクションで戻される番号は、配置テンプレートを管理するために Oracle で内部的に使用されます。

構文

```
DBMS_REPCAT_RGT.CREATE_REFRESH_TEMPLATE (  
  owner                IN   VARCHAR2,  
  refresh_group_name   IN   VARCHAR2,  
  refresh_template_name IN   VARCHAR2,  
  template_comment     IN   VARCHAR2 := NULL,  
  public_template      IN   VARCHAR2 := NULL,  
  last_modified        IN   DATE      := SYSDATE,  
  modified_by          IN   VARCHAR2 := USER,  
  creation_date        IN   DATE      := SYSDATE,  
  created_by           IN   VARCHAR2 := USER)
```

RETURN NUMBER

## パラメータ

**表 37-21 CREATE\_REFRESH\_TEMPLATE ファンクションのパラメータ**

パラメータ	説明
owner	配置テンプレート所有者のユーザー名。指定しないと、テンプレートの作成ユーザーが自動的に使用されます。
refresh_group_name	このテンプレートのインスタンス化時に作成されるリフレッシュ・グループの名前。このテンプレートで作成されたオブジェクトはすべて、指定したリフレッシュ・グループに割り当てられます。
refresh_template_name	作成する配置テンプレートの名前。この名前は、この配置テンプレートを含むすべてのアクティビティで参照されます。
template_comment	ユーザー・コメント。このコメントは、DBA_REPCAT_REFRESH_TEMPLATES ビューに表示されます。
public_template	配置テンプレートがパブリックかプライベートかを指定します。指定できる値は 'Y' と 'N' ('Y' = パブリック、'N' = プライベート) のみです。
last_modified	この配置テンプレートの最終更新日付。値を指定しないと、現行の日付が自動的に使用されます。
modified_by	この配置テンプレートの最終更新ユーザーの名前。値を指定しないと、現行ユーザーが自動的に使用されます。
creation_date	この配置テンプレートの作成日付。値を指定しないと、現行の日付が自動的に使用されます。
created_by	この配置テンプレートの作成ユーザーの名前。値を指定しないと、現行ユーザーが自動的に使用されます。

## 例外

**表 37-22 CREATE\_REFRESH\_TEMPLATE ファンクションの例外**

例外	説明
dupl_refresh_template	指定した名前のテンプレートがすでに存在します。既存のテンプレートのリストは、DBA_REPCAT_REFRESH_TEMPLATES ビューを参照してください。
bad_public_template	public_template パラメータの指定に誤りがあります。public_template パラメータには、'Y' (パブリック・テンプレート) または 'N' (プライベート・テンプレート) のいずれかを指定してください。

戻り値

表 37-23 CREATE\_REFRESH\_TEMPLATE ファンクションの戻り値

戻り値	説明
<システム生成番号>	システム生成番号は、Oracle で内部的に使用されます。

CREATE\_TEMPLATE\_OBJECT ファンクション

このファンクションは、ターゲット配置テンプレートのコンテナにオブジェクト定義を追加します。指定したオブジェクト DDL は、ターゲット配置テンプレートがリモート・スナップショット・サイトでインスタンス化されるときに実行されます。スナップショット以外に、表、プロシージャおよびその他のオブジェクトをテンプレートに追加できます。このファンクションで戻される番号は、配置テンプレートを管理するために Oracle で内部的に使用されます。

構文

```
DBMS_REPCAT_RGT.CREATE_TEMPLATE_OBJECT (  
    refresh_template_name IN VARCHAR2,  
    object_name            IN VARCHAR2,  
    object_type            IN VARCHAR2,  
    ddl_text               IN CLOB,  
    master_rollback_seg    IN VARCHAR2 := NULL,  
    flavor_id              IN NUMBER   := -1e-130)  
RETURN NUMBER;
```

パラメータ

表 37-24 CREATE\_TEMPLATE\_OBJECT ファンクションのパラメータ

パラメータ	説明
refresh_template_name	このオブジェクトを追加する配置テンプレートの名前。
object_name	作成するテンプレート・オブジェクトの名前。
object_type	テンプレートに追加するデータベース・オブジェクトの型 (SNAPSHOT、TRIGGER、PROCEDURE など)。次の型のオブジェクトを指定できます。 <div><div>SNAPSHOT</div><div>PROCEDURE</div><div>INDEX</div><div>FUNCTION</div><div>TABLE</div><div>PACKAGE</div><div>VIEW</div><div>PACKAGE BODY</div><div>SYNONYM</div><div>MATERIALIZED VIEW</div><div>SEQUENCE</div><div>DATABASE LINK</div><div>TRIGGER</div></div>
ddl_text	テンプレートに追加するオブジェクトを作成する DDL。DDL は必ずセミコロンで終了してください。(テンプレート・オブジェクトのテンプレート・パラメータの作成にはコロン(:)を使用できます。詳細は、『Oracle8i レプリケーション・ガイド』の「配置テンプレートでのスナップショットの作成」を参照してください。)
master_rollback_seg	定義済みのオブジェクト DDL をリモート・スナップショット・サイトで実行するときに使用するロールバック・セグメントの名前。
flavor_id	このテンプレート・オブジェクトの固有 ID を定義します。

例外

表 37-25 CREATE\_TEMPLATE\_OBJECT ファンクションの例外

例外	説明
miss_refresh_template	指定したリフレッシュ・テンプレート名が無効か、または存在しません。既存の配置テンプレートの一覧は、DBA_REPCAT_REFRESH_TEMPLATE ビューを参照してください。
bad_object_type	オブジェクト型の指定に誤りがあります。有効なオブジェクト型のリストは、表 37-24 を参照してください。
dupl_template_object	同じ名前と型のオブジェクトが、指定した配置テンプレートにすでに追加されています。

戻り値

表 37-26 CREATE\_TEMPLATE\_OBJECT ファンクションの戻り値

戻り値	説明
< システム生成番号 >	システム生成番号は、Oracle で内部的に使用されます。

使用上の注意

CREATE\_TEMPLATE\_OBJECT は CLOB を利用するため、このファンクションを使用するときは、DBMS\_LOB パッケージを利用する必要があります。次の例は、CREATE\_TEMPLATE\_OBJECT ファンクションで DBMS\_LOB パッケージを使用する方法を示しています。

```
DECLARE
    tempstring VARCHAR2(100);
    templob CLOB;
    a NUMBER;
BEGIN
    DBMS_LOB.CREATETEMPORARY(templob, TRUE, DBMS_LOB.SESSION);
    tempstring := 'CREATE SNAPSHOT snap_sales AS SELECT *
        FROM sales WHERE salesperson = :salesid';
    DBMS_LOB.WRITE(templob, length(tempstring), 1, tempstring);
    a := DBMS_REPCAT_RGT.CREATE_TEMPLATE_OBJECT(
        refresh_template_name => 'rgt_personnel',
        object_name => 'snap_sales',
        object_type => 'SNAPSHOT',
        ddl_text => templob,
        master_rollback_seg => 'RBS');
    DBMS_LOB.FREETEMPORARY(templob);
END;
/
```

CREATE\_TEMPLATE\_PARM ファンクション

このファンクションは、特定の配置テンプレートのパラメータを作成します。この結果、リモート・スナップショット・サイトでカスタム・データ・セットを作成できます。このファンクションは、DBA がテンプレート・オブジェクトの追加前に一連のテンプレート変数を定義するときのみ必要です（オブジェクトが CREATE\_TEMPLATE\_OBJECT ファンクションを使用してテンプレートに追加されるときに、オブジェクト DDL 内の変数が DBA\_REPCAT\_TEMPLATE\_PARS ビューに自動的に追加されます）。

通常、デフォルトのパラメータ値やプロンプト文字列を変更するときは、DBA は ALTER\_TEMPLATE\_PARM ファンクションを使用します（詳細は、[「ALTER\\_TEMPLATE\\_PARM プロシージャ」](#)（37-9 ページ）を参照してください）。このファンクションで戻される番号は、配置テンプレートを管理するために Oracle で内部的に使用されます。



構文

```
DBMS_REPCAT_RGT.CREATE_TEMPLATE_PARM (  
    refresh_template_name IN VARCHAR2,  
    parameter_name        IN VARCHAR2,  
    default_parm_value    IN CLOB      := NULL,  
    prompt_string         IN VARCHAR2 := NULL,  
    user_override         IN VARCHAR2 := NULL)  
RETURN NUMBER;
```

パラメータ

表 37-27 CREATE\_TEMPLATE\_PARM ファンクションのパラメータ

パラメータ	説明
refresh_template_name	パラメータを作成する配置テンプレートの名前。
parameter_name	作成するパラメータの名前。
default_parm_value	作成するパラメータのデフォルト値。ユーザー・パラメータ値またはランタイム・パラメータ値が存在しない場合は、インスタンス化プロセス時にこのデフォルト値が使用されます。
prompt_string	インスタンス化プロセス時にこのテンプレート・パラメータに対して表示される説明的なプロンプト・テキスト。
user_override	インスタンス化プロセス時にプロンプトが表示された場合に、ユーザーにデフォルト値の上書きを許可するかどうかを決定します（該当のパラメータに対してユーザー・パラメータ値が定義されていないと、プロンプトが表示されます）。デフォルト値の上書きを許可する場合は 'Y'、許可しない場合は 'N' を設定します。

例外

表 37-28 CREATE\_TEMPLATE\_PARM ファンクションの例外

例外	説明
miss_refresh_template	指定したリフレッシュ・テンプレート名が無効か、または存在しません。
dupl_template_parm	同じ名前のパラメータが、指定した配置テンプレートにすでに定義されています。

戻り値

表 37-29 CREATE\_TEMPLATE\_PARM ファンクションの戻り値

戻り値	説明
< システム生成番号 >	システム生成番号は、Oracle で内部的に使用されます。

使用上の注意

CREATE\_TEMPLATE\_PARM ファンクションは CLOB を利用するため、このファンクションを使用するときは、DBMS\_LOB パッケージを利用する必要があります。次の例は、CREATE\_TEMPLATE\_PARM ファンクションで DBMS\_LOB パッケージを使用する方法を示しています。

```
DECLARE
    tempstring VARCHAR2(100);
    templob CLOB;
    a NUMBER;
BEGIN
    DBMS_LOB.CREATETEMPORARY(templob, TRUE, DBMS_LOB.SESSION);
    tempstring := 'REGION 20';
    DBMS_LOB.WRITE(templob, length(tempstring), 1, tempstring);
    a := DBMS_REPCAT_RGT.CREATE_TEMPLATE_PARM(
        refresh_template_name => 'rgt_personnel',
        parameter_name => 'region',
        default_parm_value => templob,
        prompt_string => 'Enter your region ID:',
        user_override => 'Y');
    DBMS_LOB.FREETEMPORARY(templob);
END;
/
```

CREATE\_USER\_AUTHORIZATION ファンクション

このファンクションは、特定のユーザーに、プライベート配置テンプレートのインスタンス化を認可します。プライベート配置テンプレートに対する認可を付与されていないユーザーは、プライベート・テンプレートをインスタンス化できません。このファンクションは、DBA\_REPCAT\_AUTH\_TEMPLATES ビューに行を追加します。

ユーザーを認可する前に、配置テンプレートをインスタンス化するマスター・サイトに、そのユーザーが存在していることを検証してください。このファンクションで戻される番号は、配置テンプレートを管理するために Oracle で内部的に使用されます。

構文

```
DBMS_REPCAT_RGT.CREATE_USER_AUTHORIZATION (  
    user_name          IN   VARCHAR2,  
    refresh_template_name  IN   VARCHAR2)  
RETURN NUMBER;
```

パラメータ

表 37-30 CREATE\_USER\_AUTHORIZATION ファンクションのパラメータ

パラメータ	説明
user_name	指定したテンプレートのインスタンス化を認可するユーザーの名前。 複数のユーザーを指定するときは、ユーザー名をカンマで区切ります (例: 'john, mike, bob')。
refresh_template_name	指定したユーザーにインスタンス化を認可するテンプレートの名前。

例外

表 37-31 CREATE\_USER\_AUTHORIZATION ファンクションの例外

例外	説明
miss_user	指定したユーザー名が無効か、または存在しません。
miss_refresh_template	指定したリフレッシュ・テンプレート名が無効か、または存在しません。
dupl_user_ authorization	指定したユーザーと配置テンプレートに対する認可はすでに作成されています。テンプレート認可のリストは、DBA_REPCAT_AUTH_TEMPLATES ビューを参照してください。

戻り値

表 37-32 CREATE\_USER\_AUTHORIZATION ファンクションの戻り値

戻り値	説明
< システム生成番号 >	システム生成番号は、Oracle で内部的に使用されます。

## CREATE\_USER\_PARM\_VALUE ファンクション

このファンクションは、特定のユーザーに対する配置テンプレートのパラメータ値を事前定義します。たとえば、ユーザー 33456 のリージョン・パラメータを WEST に再定義する場合などに、このファンクションを使用します。

このファンクションで指定した値は、テンプレート・パラメータに対して指定したデフォルト値よりも優先されます。このファンクションで戻される番号は、配置テンプレートを管理するために Oracle で内部的に使用されます。

### 構文

```
DBMS_REPCAT_RGT.CREATE_USER_PARM_VALUE (  
  refresh_template_name    IN   VARCHAR2,  
  parameter_name           IN   VARCHAR2,  
  user_name                IN   VARCHAR2,  
  parm_value               IN   CLOB := NULL)  
RETURN NUMBER;
```

### パラメータ

表 37-33 CREATE\_USER\_PARM\_VALUE ファンクションのパラメータ

パラメータ	説明
refresh_template_name	ユーザー値を作成するパラメータを含んだ配置テンプレートの名前。
parameter_name	ユーザー・パラメータ値を定義するテンプレート・パラメータの名前。
user_name	パラメータ値を事前定義の対象とするユーザーの名前。
parm_value	事前定義パラメータ値。指定したユーザーによって開始されたインスタンス化プロセス時に使用されます。

例外

表 37-34 CREATE\_USER\_PARM\_VALUE ファンクションの例外

例外	説明
miss_refresh_template	指定した配置テンプレート名が無効か、または存在しません。
dupl_user_parm_values	指定したユーザー、パラメータおよび配置テンプレートに対するパラメータ値はすでに定義されています。既存のユーザー・パラメータ値のリストは、DBA_REPCAT_USER_PARS ビューを参照してください。
miss_template_parm	指定した配置テンプレート・パラメータ名が無効か、または存在しません。
miss_user	指定したユーザー名が無効か、または存在しません。

戻り値

表 37-35 CREATE\_USER\_PARM\_VALUE ファンクションの戻り値

戻り値	説明
< システム生成番号 >	システム生成番号は、Oracle で内部的に使用されます。

使用上の注意

CREATE\_USER\_PARM\_VALUE ファンクションは CLOB を利用するため、このファンクションを使用するときは、DBMS\_LOB パッケージを利用する必要があります。次の例は、CREATE\_USER\_PARM\_VALUE ファンクションで DBMS\_LOB パッケージを使用する方法を示しています。

```
DECLARE
    tempstring VARCHAR2(100);
    templob CLOB;
    a NUMBER;
BEGIN
    DBMS_LOB.CREATETEMPORARY(templob, TRUE, DBMS_LOB.SESSION);
    tempstring := 'REGION 20';
    DBMS_LOB.WRITE(templob, length(tempstring), 1, tempstring);
    a := DBMS_REPCAT_RGT.CREATE_USER_PARM_VALUE(
        refresh_template_name => 'rgt_personnel',
        parameter_name => 'region',
        user_name => 'BOB',
        user_parm_value => templob);
    DBMS_LOB.FREETEMPORARY(templob);
END;
```

## DELETE\_RUNTIME\_PARMS プロシージャ

このプロシージャは、配置テンプレートをインスタンス化する前に、INSERT\_RUNTIME\_PARMS プロシージャを使用して定義したランタイム・パラメータ値を削除するために使用します。

### 構文

```
DBMS_REPCAT_RGT.DELETE_RUNTIME_PARMS(  
    runtime_parm_id    IN    NUMBER,  
    parameter_name     IN    VARCHAR2);
```

### パラメータ

表 37-36 DELETE\_RUNTIME\_PARMS プロシージャのパラメータ

パラメータ	説明
runtime_parm_id	以前にランタイム・パラメータ値に割り当てた ID (GET_RUNTIME_PARM_ID ファンクションを使用して取り出された値です) を指定します。
parameter_name	削除するパラメータ値の名前 (配置テンプレート・パラメータのリストは、DBA_REPCAT_TEMPLATE_PARMS を参照してください) を指定します。

### 例外

表 37-37 DELETE\_RUNTIME\_PARMS プロシージャの例外

例外	説明
miss_template_parm	指定した配置テンプレート・パラメータ名が無効か、または存在しません。

## DROP\_ALL\_OBJECTS プロシージャ

このプロシージャでは、DBA が配置テンプレートからすべてのオブジェクトまたは特定のオブジェクト型を削除できます。

**注意：** このプロシージャは、元に戻すことのできない危険度の高いプロシージャです。

構文

```
DBMS_REPCAT_RGT.DROP_ALL_OBJECTS (  
    refresh_template_name    IN    VARCHAR2,  
    object_type              IN    VARCHAR2 := NULL);
```

パラメータ

表 37-38 DROP\_ALL\_OBJECTS プロシージャのパラメータ

パラメータ	説明
refresh_template_name	削除するオブジェクトを含んだ配置テンプレートの名前。
object_type	NULL を指定すると、テンプレート内のすべてのオブジェクトが削除されます。オブジェクト型を指定すると、その型のオブジェクトのみ削除されます。次の型のオブジェクトを指定できます。 <div><div>SNAPSHOT</div><div>PROCEDURE</div><div>INDEX</div><div>FUNCTION</div><div>TABLE</div><div>PACKAGE</div><div>VIEW</div><div>PACKAGE BODY</div><div>SYNONYM</div><div>MATERIALIZED VIEW</div><div>SEQUENCE</div><div>DATABASE LINK</div><div>TRIGGER</div></div>

例外

表 37-39 DROP\_ALL\_OBJECTS プロシージャの例外

例外	説明
miss_refresh_template	指定した配置テンプレート名が無効か、または存在しません。
bad_object_type	オブジェクト型の指定に誤りがあります。有効なオブジェクト型のリストは、 <a href="#">表 37-38</a> を参照してください。

DROP\_ALL\_TEMPLATE\_PARMS プロシージャ

このプロシージャでは、DBA が、指定した配置テンプレートのテンプレート・パラメータを削除できます。テンプレート・オブジェクトで参照されていないすべてのパラメータ、またはパラメータを参照するすべてのオブジェクトとそのパラメータ自体を削除できます。

**注意： このプロシージャは、元に戻すことのできない危険度の高いプロシージャです。**

構文

```
DBMS_REPCAT_RGT.DROP_ALL_TEMPLATE_PARMS (  
    refresh_template_name    IN    VARCHAR2,  
    drop_objects              IN    VARCHAR2 := N);
```

パラメータ

表 37-40 DROP\_ALL\_TEMPLATE\_PARMS プロシージャのパラメータ

パラメータ	説明
refresh_template_name	削除するパラメータを含んだ配置テンプレートの名前。
drop_objects	値を指定しないと、デフォルトで N に設定され、テンプレート・オブジェクトで参照されていないパラメータがすべて削除されます。  Y を指定すると、テンプレート・パラメータを参照するすべてのオブジェクトとそのテンプレート・パラメータ自体が削除されます。

例外

表 37-41 DROP\_ALL\_TEMPLATE\_PARMS プロシージャの例外

例外	説明
miss_refresh_template	指定した配置テンプレート名が無効か、または存在しません。

DROP\_ALL\_TEMPLATE\_SITES プロシージャ

このプロシージャは、DBA\_REPCAT\_TEMPLATE\_SITES ビューからエントリをすべて削除します。このビューには、特定の配置テンプレートをインスタンス化したサイトの記録が格納されています。

**注意： このプロシージャは、元に戻すことのできない危険度の高いプロシージャです。また、削除されたテンプレートをインスタンス化している Oracle Lite のサイトでは、そのスナップショットをリフレッシュできなくなります。**



構文

```
DBMS_REPCAT_RGT.DROP_ALL_TEMPLATE_SITES (  
    refresh_template_name IN VARCHAR2);
```

パラメータ

表 37-42 DROP\_ALL\_TEMPLATE\_SITES プロシージャのパラメータ

パラメータ	説明
refresh_template_name	削除するサイトを含んだ配置テンプレートの名前。

例外

表 37-43 DROP\_ALL\_TEMPLATE\_SITES プロシージャの例外

例外	説明
miss_refresh_template	指定した配置テンプレート名が無効か、または存在しません。

DROP\_ALL\_TEMPLATES プロシージャ

このプロシージャは、プロシージャがコールされるサイトの配置テンプレートをすべて削除します。

**注意：** このプロシージャは、元に戻すことのできない危険度の高いプロシージャです。

構文

```
DBMS_REPCAT_RGT.DROP_ALL_TEMPLATES;
```

パラメータ

なし。

DROP\_ALL\_USER\_AUTHORIZATIONS プロシージャ

このプロシージャでは、DBA が、指定した配置テンプレートに対するユーザー認証をすべて削除できます。このプロシージャを実行すると、DBA\_REPCAT\_AUTH\_TEMPLATES ビューから行が削除されます。

このプロシージャは、プライベート・テンプレートがパブリック・テンプレートに変換され、ユーザー認証が不要になってからインプリメントされることがあります。

構文

```
DBMS_REPCAT_RGT.DROP_ALL_USER_AUTHORIZATIONS (  
    refresh_template_name IN VARCHAR2);
```

パラメータ

表 37-44 DROP\_ALL\_USER\_AUTHORIZATIONS プロシージャのパラメータ

パラメータ	説明
refresh_template_name	削除するオブジェクトを含んだ配置テンプレートの名前。

例外

表 37-45 DROP\_ALL\_USER\_AUTHORIZATIONS プロシージャの例外

例外	説明
miss_refresh_template	指定した配置テンプレート名が無効か、または存在しません。

DROP\_ALL\_USER\_PARM\_VALUES プロシージャ

このプロシージャは、特定の配置テンプレートに対するユーザー・パラメータ値を削除します。このプロシージャは柔軟に設計されており、DBA が、削除するユーザー・パラメータ値のセットを定義できます。たとえば、パラメータの指定方法によって次のように処理されます。

- refresh\_template\_name: 指定した配置テンプレートに対するユーザー・パラメータをすべて削除します。
- refresh\_template\_name、user\_name: 指定した配置テンプレートに対する指定したユーザー・パラメータをすべて削除します。
- refresh\_template\_name、parameter\_name: 指定した配置テンプレート・パラメータに対するユーザー・パラメータ値をすべて削除します。
- refresh\_template\_name、parameter\_name、user\_name: 指定した配置テンプレート・パラメータに対する指定したユーザー値を削除します ( DROP\_USER\_PARM と同じです )。

構文

```
DBMS_REPCAT_RGT.DROP_ALL_USER_PARMS (  
    refresh_template_name IN VARCHAR2,  
    user_name              IN VARCHAR2,  
    parameter_name         IN VARCHAR2);
```

## パラメータ

表 37-46 DROP\_ALL\_USER\_PARMS プロシージャのパラメータ

パラメータ	説明
refresh_template_name	削除するパラメータ値を含んだ配置テンプレートの名前。
user_name	パラメータ値を削除対象とするユーザーの名前。
parameter_name	削除する値を含んだテンプレート・パラメータ。

## 例外

表 37-47 DROP\_ALL\_USER\_PARMS プロシージャの例外

例外	説明
miss_refresh_template	指定した配置テンプレート名が無効か、または存在しません。
miss_user	指定したユーザー名が無効か、または存在しません。
miss_user_parm_values	配置テンプレート、ユーザーおよびパラメータの組合せが DBA_REPCAT_USER_PARM ビューに存在しません。

## DROP\_REFRESH\_TEMPLATE プロシージャ

このプロシージャは配置テンプレートを削除します。配置テンプレートを削除すると、関連するすべてのテンプレート・パラメータ、ユーザー認証、テンプレート・オブジェクトおよびユーザー・パラメータも段階的に削除されます（このプロシージャでは、テンプレート・サイトは削除されません）。

## 構文

```
DBMS_REPCAT_RGT.DROP_REFRESH_TEMPLATE (  
    refresh_template_name IN VARCHAR2);
```

## パラメータ

表 37-48 DROP\_REFRESH\_TEMPLATE プロシージャのパラメータ

パラメータ	説明
refresh_template_name	削除する配置テンプレート名。

例外

表 37-49 DROP\_REFRESH\_TEMPLATE プロシージャの例外

例外	説明
miss_refresh_template	指定した配置テンプレート名が無効か、または存在しません。配置テンプレートのリストは、DBA_REPCAT_REFRESH_TEMPLATE ビューを参照してください。

DROP\_SITE\_INSTANTIATION プロシージャ

用途

このプロシージャは、ターゲット・サイトのテンプレート・インスタンスエーションを削除します。また、マスター・サイトの関連メタデータをすべて削除し、指定したサイトにおけるスナップショットのリフレッシュを使用禁止にします。

構文

```
DBMS_REPCAT_RGT.DROP_SITE_INSTANTIATION(  
  refresh_template_name  IN   VARCHAR2,  
  user_name              IN   VARCHAR2,  
  site_name              IN   VARCHAR2,  
  repapi_site_id         IN   NUMBER := -1e-130);
```

パラメータ

表 37-50 DROP\_SITE\_INSTANTIATION のパラメータ

パラメータ	説明
refresh_template_name	削除する配置テンプレートの名前。
user_name	リモート・スナップショット・サイトでそのテンプレートを最初にインスタンス化したユーザーの名前。テンプレートをインスタンス化したユーザーを調べるには、REPCAT_TEMPLATE_SITES ビューを参照してください。
site_name	指定したテンプレート・インスタンスエーションを削除する Oracle Server のサイトを示します。( site_name を指定した場合、 repapi_site_id は指定しないでください。)
repapi_site_id	指定したテンプレート・インスタンスエーションを削除する REPAPI の位置を示します。( repapi_site_id を指定した場合、 site_name は指定しないでください。)

# DROP\_TEMPLATE\_OBJECT プロシージャ

このプロシージャは、特定の配置テンプレートからテンプレート・オブジェクトを削除します。たとえば、DBA は、古いスナップショットを配置テンプレートから削除するためにこのプロシージャを使用できます。テンプレートの変更内容は、変更後に配置テンプレートをインスタンス化する新規サイトに反映されます。テンプレートをすでにインスタンス化しているリモート・サイトは、配置テンプレートを再インスタンス化して変更内容を適用する必要があります。

## 構文

```
DBMS_REPCAT_RGT.DROP_TEMPLATE_OBJECT (  
    refresh_template_name  IN   VARCHAR2,  
    object_name            IN   VARCHAR2,  
    object_type            IN   VARCHAR2);
```

## パラメータ

表 37-51 DROP\_TEMPLATE\_OBJECT プロシージャのパラメータ

パラメータ	説明
refresh_template_name	オブジェクトを削除する配置テンプレートの名前。
object_name	削除するテンプレート・オブジェクトの名前。
object_type	削除するオブジェクトの型。次の型のオブジェクトを指定できます。 <div><div>SNAPSHOT</div><div>PROCEDURE</div><div>INDEX</div><div>FUNCTION</div><div>TABLE</div><div>PACKAGE</div><div>VIEW</div><div>PACKAGE BODY</div><div>SYNONYM</div><div>MATERIALIZED VIEW</div><div>SEQUENCE</div><div>DATABASE LINK</div><div>TRIGGER</div></div>

## 例外

表 37-52 DROP\_TEMPLATE\_OBJECT プロシージャの例外

例外	説明
miss_refresh_template	指定した配置テンプレート名が無効か、または存在しません。
miss_template_object	指定したテンプレート・オブジェクトが無効か、または存在しません。配置テンプレート・オブジェクトのリストは、DBA_REPCAT_TEMPLATE_OBJECT ビューを参照してください。

## DROP\_TEMPLATE\_PARM プロシージャ

このプロシージャは、DBA\_REPCAT\_TEMPLATE\_PARMS ビューから既存のテンプレート・パラメータを削除します。あるテンプレート・オブジェクトを削除して、特定のパラメータが不要になった場合に有効なプロシージャです。

### 構文

```
DBMS_REPCAT_RGT.DROP_TEMPLATE_PARM (  
    refresh_template_name IN VARCHAR2,  
    parameter_name        IN VARCHAR2);
```

### パラメータ

表 37-53 DROP\_TEMPLATE\_PARM プロシージャのパラメータ

パラメータ	説明
refresh_template_name	削除するパラメータを含んだ配置テンプレートの名前。
parameter_name	削除するパラメータの名前。

### 例外

表 37-54 DROP\_TEMPLATE\_PARM プロシージャの例外

例外	説明
miss_refresh_template	指定した配置テンプレート名が無効か、または存在しません。
miss_template_parm	指定したパラメータ名が無効か、または存在しません。テンプレート・パラメータのリストは、DBA_REPCAT_TEMPLATE_PARMS ビューを参照してください。

## DROP\_USER\_AUTHORIZATION プロシージャ

このプロシージャは、DBA\_REPCAT\_TEMPLATE\_AUTH ビューからユーザー認証エントリを削除します。ユーザーのテンプレート認可を削除するときに使用します。ユーザーの認証が削除されると、そのユーザーはターゲット配置テンプレートをインスタンス化できません。

関連項目：「[DROP\\_ALL\\_USER\\_AUTHORIZATIONS プロシージャ](#)」  
( 37-31 ページ )

### 構文

```
DBMS_REPCAT_RGT.DROP_USER_AUTHORIZATION (  
    refresh_template_name IN VARCHAR2,  
    user_name             IN VARCHAR2);
```

## パラメータ

**表 37-55 DROP\_USER\_AUTHORIZATION プロシージャのパラメータ**

パラメータ	説明
refresh_template_name	ユーザーの認証を削除する配置テンプレートの名前。
user_name	認証を削除対象とするユーザーの名前。

## 例外

**表 37-56 DROP\_USER\_AUTHORIZATION プロシージャの例外**

例外	説明
miss_user	指定したユーザー名が無効か、または存在しません。
miss_user_ authorization	指定したユーザーと配置テンプレートの組合せが存在しません。ユーザーまたは配置テンプレートのリストは、DBA_REPCAT_TEMPLATE_AUTH を参照してください。
miss_refresh_template	指定した配置テンプレート名が無効か、または存在しません。

## DROP\_USER\_PARM\_VALUE プロシージャ

このプロシージャは、特定の配置テンプレートに対する事前定義ユーザー・パラメータ値を削除します。通常、ユーザーのテンプレート認可を削除した後に実行します。

## 構文

```
DBMS_REPCAT_RGT.DROP_USER_PARM_VALUE (  
    refresh_template_name    IN    VARCHAR2,  
    parameter_name          IN    VARCHAR2,  
    user_name                IN    VARCHAR2);
```

## パラメータ

**表 37-57 DROP\_USER\_PARM\_VALUE プロシージャのパラメータ**

パラメータ	説明
refresh_template_name	削除するパラメータ値を含んだ配置テンプレート名。
parameter_name	削除する事前定義値を含んだテンプレート名。
user_name	パラメータ値を削除対象とするユーザーの名前。

例外

表 37-58 DROP\_USER\_PARM\_VALUE プロシージャの例外

例外	説明
miss_refresh_ template	指定した配置テンプレート名が無効か、または存在しません。
miss_user	指定したユーザー名が無効か、または存在しません。
miss_user_parm_ values	配置テンプレート、ユーザーおよびパラメータの組合せが DBA_ REPCAT_USER_PARM ビューに存在しません。

GET\_RUNTIME\_PARM\_ID ファンクション

このファンクションは、ランタイム・パラメータ値の定義時に使用する ID を取り出します。  
ランタイム・パラメータ値はすべてこの ID に割り当てられ、インスタンス化プロセス時にも使用されます。

構文

```
DBMS_REPCAT_RGT.GET_RUNTIME_PARM_ID  
RETURN NUMBER;
```

パラメータ

なし。

戻り値

表 37-59 GET\_RUNTIME\_PARM\_ID ファンクションの戻り値

戻り値	対応するデータ型
< システム生成番号 >	ランタイム・パラメータ値はこのシステム生成番号に割り当てられ、 インスタンス化プロセス時にも使用されます。



## INSERT\_RUNTIME\_PARMS プロシージャ

このプロシージャは、テンプレートのインスタンス化前にランタイム・パラメータ値を定義します。ユーザー・パラメータ値が未定義で、デフォルトのパラメータ値を使用しない場合は、このプロシージャを使用してパラメータ値を定義する必要があります。

このプロシージャを使用する前に、必ず GET\_RUNTIME\_PARM\_ID ファンクションを実行して、ランタイム・パラメータの挿入時に使用するパラメータ ID を取り出してください。この ID は、ランタイム・パラメータ値の定義と配置テンプレートのインスタンス化に使用されます。

### 構文

```
DBMS_REPCAT_RGT.INSERT_RUNTIME_PARMS (
    runtime_parm_id    IN    NUMBER,
    parameter_name     IN    VARCHAR2,
    parameter_value    IN    CLOB);
```

### パラメータ

表 37-60 INSERT\_RUNTIME\_PARMS プロシージャのパラメータ

パラメータ	説明
runtime_parm_id	GET_RUNTIME_PARM_ID ファンクションで取り出された ID。この ID は、配置テンプレートのインスタンス化時にも使用されます（配置テンプレートのすべてのパラメータ値には必ず同じ ID を使用してください）。
parameter_name	ランタイム・パラメータ値を定義するテンプレート・パラメータの名前（テンプレート・パラメータのリストは、DBA_REPCAT_TEMPLATE_PARMS ビューを参照してください）。
parameter_value	配置テンプレートのインスタンス化プロセス時に使用するランタイム・パラメータ値。

### 例外

表 37-61 INSERT\_RUNTIME\_PARMS プロシージャの例外

例外	説明
miss_refresh_template	指定した配置テンプレート名が無効か、または存在しません。
miss_user	指定したユーザー名が無効か、または存在しません。
miss_user_parm_values	配置テンプレート、ユーザーおよびパラメータの組合せが DBA_REPCAT_USER_PARM ビューに存在しません。

## 使用上の注意

INSERT\_RUNTIME\_PARMS は、CLOB を利用するため、このプロシージャを使用する場合は、DBMS\_LOB パッケージを利用する必要があります。次の例は、INSERT\_RUNTIME\_PARMS プロシージャで DBMS\_LOB パッケージを使用する方法を示しています。

```
DECLARE
    tempstring VARCHAR2(100);
    templob CLOB;
BEGIN
    DBMS_LOB.CREATETEMPORARY(templob, TRUE, DBMS_LOB.SESSION);
    tempstring := 'REGION 20';
    DBMS_LOB.WRITE(templob, length(tempstring), 1, tempstring);
    DBMS_REPCAT_RGT.INSERT_RUNTIME_PARMS(
        runtime_parm_id => 20,
        parameter_name => 'region',
        parameter_value => templob);
    DBMS_LOB.FREETEMPORARY(templob);
END;
/
```

## INstantiate\_Offline ファンクション

このファンクションは、マスター・サイトでスクリプトを生成します。このスクリプトは、オフライン時にリモート・スナップショット・サイトでスナップショット環境を作成するために使用します。リモート・スナップショット・サイトでのインスタンス化プロセスは長くなる場合があるため（必要な時間は新しいスナップショットに移入されるデータの量によって異なります）生成されたスクリプトは、マスター・サイトに長時間接続したままにできるリモート・スナップショット・サイトで使用してください。このプロシージャは、ユーザー・インスタンスエーションごとに別々に実行する必要があります。

このファンクションで生成されたスクリプトは、USER\_REPCAT\_TEMP\_OUTPUT 一時ビューに格納され、配置テンプレートの配布時に、Replication Manager などの Oracle Tools で使用されます。このファンクションで戻される数値は、USER\_REPCAT\_TEMP\_OUTPUT ビューから該当する情報を取り出すために使用されます。

**注意：** このファンクションは、配置テンプレートのオフライン・インスタンス化の実行時に使用されます。また、別のユーザーのためにインスタンス化を実行しているレプリケーション管理者用でもあります。独自のインスタンス化を実行するユーザーは、パブリック・バージョンの「[INSTANTIATE\\_OFFLINE ファンクション](#)」( 36-3 ページ ) を使用してください。

このファンクションを、[DBMS\\_OFFLINE\\_OG](#) パッケージ内のプロシージャ ( マスター表のオフライン・インスタンス化の実行に使用 ) や [DBMS\\_OFFLINE\\_SNAPSHOT](#) パッケージ内のプロシージャ ( スナップショットのオフライン・インスタンス化の実行に使用 ) と混同しないでください。これらの使用方法の詳細は、各パッケージの説明を参照してください。

構文

```
DBMS_REPCAT_RGT.INSTANTIATE_OFFLINE(  
    refresh_template_name    IN    VARCHAR2,  
    site_name                IN    VARCHAR2,  
    user_name                IN    VARCHAR2 := NULL,  
    runtime_parm_id          IN    NUMBER   := -1e-130,  
    next_date                IN    DATE     := SYSDATE,  
    interval                 IN    VARCHAR2 : 'SYSDATE + 1')  
RETURN NUMBER;
```

パラメータ

表 37-62 INSTANTIATE\_OFFLINE ファンクションのパラメータ

パラメータ	説明
refresh_template_name	インスタンス化する配置テンプレートの名前。
site_name	配置テンプレートをインスタンス化するリモート・サイトの名前。
user_name	配置テンプレートをインスタンス化する認可ユーザーの名前。
runtime_parm_id	INSERT_RUNTIME_PARMS プロシージャを使用してランタイム・パラメータ値を定義した場合は、ランタイム・パラメータの作成時に使用した ID ( GET_RUNTIME_PARM_ID ファンクションを使用して取り出された ID です ) を指定します。
next_date	リフレッシュ・グループの作成時に使用される次回リフレッシュ日付の値を指定します。
interval	リフレッシュ・グループの作成時に使用されるリフレッシュ間隔を指定します。

例外

表 37-63 INSTANTIATE\_OFFLINE ファンクションの例外

例外	説明
miss_refresh_template	指定した配置テンプレート名が無効か、または存在しません。
miss_user	指定した認可ユーザー名が無効か、または存在しません。指定したユーザーが DBA_REPCAT_TEMPLATE_AUTH ビューにリストされていることを検証してください。リストされていない場合、指定したユーザーは、ターゲット配置テンプレートのインスタンス化を認可されていません。
bad_parms	定義したユーザー・パラメータ値またはテンプレート・デフォルト値によって移入されていないテンプレート・パラメータがあります。事前定義値の数とテンプレート・パラメータ数が一致していないか、または事前定義値がターゲット・パラメータに対して無効です（型の不一致など）。

戻り値

表 37-64 INSTANTIATE\_OFFLINE ファンクションの戻り値

戻り値	説明
< システム生成番号 >	システムによって生成された番号。生成されたインスタンスエーション・スクリプトを取り出すため USER_REPCAT_TEMP_OUTPUT ビューから選択するときに、output_id に使用する値です。

INSTANTIATE\_ONLINE ファンクション

このファンクションは、マスター・サイトでスクリプトを生成します。このスクリプトは、オンライン時にリモート・スナップショット・サイトでスナップショット環境を作成するために使用します。リモート・スナップショット・サイトでのインスタンス化プロセスは長くなる場合があるため（必要な時間は新しいスナップショットに移入されるデータの量によって異なります）生成されたスクリプトは、マスター・サイトに長時間接続したままにできるリモート・スナップショット・サイトで使用してください。このプロシージャは、ユーザー・インスタンスエーションごとに別々に行う必要があります。

このファンクションで生成されたスクリプトは、USER\_REPCAT\_TEMP\_OUTPUT 一時ビューに格納され、配置テンプレートの配布時に、Replication Manager などの Oracle Tools で使用されます。このファンクションで戻される数値は、USER\_REPCAT\_TEMP\_OUTPUT ビューから該当する情報を取り出すために使用されます。

**注意：** このファンクションは、別のユーザーのためにインスタンス化を実行しているレプリケーション管理者用です。独自のインスタンス化を実行するユーザーは、パブリック・バージョンの「[INSTANTIATE\\_ONLINE ファンクション](#)」( 36-5 ページ ) を使用してください。

構文

```
DBMS_REPCAT_RGT.INSTANTIATE_ONLINE(  
  refresh_template_name  IN   VARCHAR2,  
  site_name              IN   VARCHAR2 := NULL,  
  user_name              IN   VARCHAR2 := NULL,  
  runtime_parm_id        IN   NUMBER   := -1e-130,  
  next_date              IN   DATE      := SYSDATE,  
  interval               IN   VARCHAR2 := 'SYSDate + 1')  
RETURN NUMBER;
```

パラメータ

表 37-65 INSTANTIATE\_ONLINE ファンクションのパラメータ

パラメータ	説明
refresh_template_name	インスタンス化する配置テンプレートの名前。
site_name	配置テンプレートをインスタンス化するリモート・サイトの名前。
user_name	配置テンプレートをインスタンス化する認可ユーザーの名前。
runtime_parm_id	INSERT_RUNTIME_PARMS プロシージャを使用してランタイム・パラメータ値を定義した場合は、ランタイム・パラメータの作成時に使用した ID ( GET_RUNTIME_PARM_ID ファンクションを使用して取り出された ID です ) を指定します。
next_date	リフレッシュ・グループの作成時に使用される次回リフレッシュ日付の値を指定します。
interval	リフレッシュ・グループの作成時に使用されるリフレッシュ間隔を指定します。

例外

表 37-66 INSTANTIATE\_ONLINE ファンクションの例外

例外	説明
miss_refresh_template	指定した配置テンプレート名が無効か、または存在しません。
miss_user	指定した認可ユーザー名が無効か、または存在しません。指定したユーザーが DBA_REPCAT_TEMPLATE_AUTH ビューにリストされていることを検証してください。リストされていない場合、指定したユーザーは、ターゲット配置テンプレートのインスタンス化を認可されていません。
bad_parms	定義したユーザー・パラメータ値またはテンプレート・デフォルト値によって移入されていないテンプレート・パラメータがあります。事前定義値の数とテンプレート・パラメータ数が一致していないか、または事前定義値がターゲット・パラメータに対して無効です（型の不一致など）。

戻り値

表 37-67 INSTANTIATE\_ONLINE ファンクションの戻り値

戻り値	説明
< システム生成番号 >	システムによって生成された番号。生成されたインスタンスエーション・スクリプトを取り出すため USER_REPCAT_TEMP_OUTPUT ビューから選択するときに、output_id に使用する値です。

LOCK\_TEMPLATE\_EXCLUSIVE プロシージャ

このプロシージャは、配置テンプレートの更新中または変更中に、ユーザーがテンプレートの読み込みまたはインスタンス化を実行できないようにします。

ロックは、ROLLBACK または COMMIT が実行されると解放されます。

**注意：** 配置テンプレートを変更する前に、このプロシージャを必ず実行してください。

構文

```
DBMS_REPCAT_RGT.LOCK_TEMPLATE_EXCLUSIVE;
```

パラメータ

なし。

## LOCK\_TEMPLATE\_SHARED プロシージャ

このプロシージャは、指定した配置テンプレートを読み取り専用にします。テンプレートをインスタンス化する前に、このプロシージャを必ずコールしてください。この結果、インスタンス化中に他のユーザーによって配置テンプレートが変更されないことが保証されます。

ロックは、ROLLBACK または COMMIT が実行されると解放されます。

### 構文

```
DBMS_REPCAT_RGT.LOCK_TEMPLATE_SHARED;
```

### パラメータ

なし。





---

## DBMS\_REPUTIL

DBMS\_REPUTIL には、表のレプリケーションで使用するシャドー表、トリガーおよびパッケージを生成するサブプログラムの他に、スタンドアロン・プロシージャ起動とパッケージ・プロシージャ起動のレプリケーションに使用するラッパーを生成するサブプログラムが含まれています。

このパッケージは、生成コードでのみ参照されます。

# サブプログラムの要約

表 38-1 DBMS\_REPUTIL パッケージのサブプログラム

サブプログラム	説明
<a href="#">REPLICATION_OFF プロシージャ</a> ( 38-2 ページ )	変更内容をレプリケート環境にある他のサイトにレプリケートせずに表を変更したり、またはプロシージャ・レプリケーションの使用時に行レベル・レプリケーションを使用禁止にします。
<a href="#">REPLICATION_ON プロシージャ</a> ( 38-2 ページ )	レプリケーションが一時的に中断された後に、変更内容のレプリケーションを再び使用可能にします。
<a href="#">REPLICATION_IS_ON ファンクション</a> ( 38-3 ページ )	レプリケーションが実行中かどうかを判別します。
<a href="#">FROM_REMOTE ファンクション</a> ( 38-3 ページ )	内部レプリケーション・パッケージにあるプロシージャの開始時に TRUE を戻し、終了時に FALSE を戻します。
<a href="#">GLOBAL_NAME ファンクション</a> ( 38-3 ページ )	ローカル・データベースのグローバル・データベース名を判別します ( 戻り値はグローバル名 )。

## REPLICATION\_OFF プロシージャ

このプロシージャは、変更内容をレプリケート環境にある他のサイトにレプリケートせずに表を変更したり、またはプロシージャ・レプリケーションの使用時に行レベル・レプリケーションを使用禁止にします。一般的に、このフラグを設定する前には、レプリケート環境にあるすべてのマスター・グループに対するレプリケーション・アクティビティを中断する必要があります。

### 構文

DBMS\_REPUTIL.REPLICATION\_OFF;

### パラメータ

なし。

## REPLICATION\_ON プロシージャ

このプロシージャは、レプリケーションが一時的に中断された後に、変更内容のレプリケーションを再び使用可能にします。

### 構文

DBMS\_REPUTIL.REPLICATION\_ON;

## パラメータ

なし。

## REPLICATION\_IS\_ON ファンクション

このファンクションは、レプリケーションが実行中かどうかを判別します。戻り値が TRUE の場合は、生成されたレプリケーション・トリガーが使用可能であることを示します。FALSE の場合は、レプリケート・オブジェクト・グループに対して、現行サイトでのレプリケーションは使用禁止であることを示します。

このファンクションの戻り値は、DBMS\_REPUTIL パッケージの REPLICATION\_ON または REPLICATION\_OFF プロシージャをコールして設定されます。

## 構文

```
DBMS_REPUTIL.REPLICATION_IS_ON  
    RETURN BOOLEAN;
```

## パラメータ

なし。

## FROM\_REMOTE ファンクション

このファンクションは、内部レプリケーション・パッケージにあるプロシージャの開始時に TRUE を戻し、終了時に FALSE を戻します。内部パッケージによる更新の結果、起動するトリガーがある場合は、このファンクションをチェックする必要があります。

## 構文

```
DBMS_REPUTIL.FROM_REMOTE  
    RETURN BOOLEAN;
```

## パラメータ

なし。

## GLOBAL\_NAME ファンクション

このファンクションは、ローカル・データベースのグローバル・データベース名を判別します（戻り値はグローバル名）。

## 構文

```
DBMS_REPUTIL.GLOBAL_NAME  
    RETURN VARCHAR2;
```

## パラメータ

なし。

---

## DBMS\_RESOURCE\_MANAGER

DBMS\_RESOURCE\_MANAGER パッケージは、プラン、コンシューマ・グループおよびプラン・ディレクティブをメンテナンスします。また、プラン・スキーマへの変更内容をグループ化する方法も提供します。

**関連項目：** データベース・リソース・マネージャの使用の詳細は、『Oracle8i 管理者ガイド』を参照してください。

要件

実行者には、このプロシージャを実行するための ADMINISTER\_RESOURCE\_MANAGER システム権限が必要です。この権限の付与と取消しを行うプロシージャは、DBMS\_RESOURCE\_MANAGER\_PRIVS パッケージにあります。

サブプログラムの要約

表 39-1 DBMS\_RESOURCE\_MANAGER パッケージのサブプログラム

サブプログラム	説明
<a href="#">CREATE_PLAN プロシージャ</a> ( 39-3 ページ )	リソース・プランを定義するエントリを作成します。
<a href="#">UPDATE_PLAN プロシージャ</a> ( 39-4 ページ )	リソース・プランを定義するエントリを更新します。
<a href="#">DELETE_PLAN プロシージャ</a> ( 39-4 ページ )	指定のプランとそれが参照するすべてのプラン・ディレクトティブを削除します。
<a href="#">DELETE_PLAN_CASCADE プロシージャ</a> ( 39-5 ページ )	指定のプランとそのすべての子 ( プラン・ディレクティブ、サブプラン、コンシューマ・グループ ) を削除します。
<a href="#">CREATE_CONSUMER_GROUP プロシージャ</a> ( 39-6 ページ )	リソース・コンシューマ・グループを定義するエントリを作成します。
<a href="#">UPDATE_CONSUMER_GROUP プロシージャ</a> ( 39-6 ページ )	リソース・コンシューマ・グループを定義するエントリを更新します。
<a href="#">DELETE_CONSUMER_GROUP プロシージャ</a> ( 39-7 ページ )	リソース・コンシューマ・グループを定義するエントリを削除します。
<a href="#">CREATE_PLAN_DIRECTIVE プロシージャ</a> ( 39-8 ページ )	リソース・プラン・ディレクティブを作成します。
<a href="#">UPDATE_PLAN_DIRECTIVE プロシージャ</a> ( 39-9 ページ )	リソース・プラン・ディレクティブを更新します。
<a href="#">DELETE_PLAN_DIRECTIVE プロシージャ</a> ( 39-10 ページ )	リソース・プラン・ディレクティブを削除します。
<a href="#">CREATE_PENDING_AREA プロシージャ</a> ( 39-10 ページ )	リソース・マネージャ・オブジェクトへの変更を行うための作業領域を作成します。
<a href="#">VALIDATE_PENDING_AREA プロシージャ</a> ( 39-12 ページ )	リソース・マネージャに対する保留中の変更を検証します。
<a href="#">CLEAR_PENDING_AREA プロシージャ</a> ( 39-12 ページ )	リソース・マネージャに対する作業領域を消去します。

表 39-1 DBMS\_RESOURCE\_MANAGER パッケージのサブプログラム

サブプログラム	説明
<a href="#">SUBMIT_PENDING_AREA プロシージャ</a> (39-12 ページ)	リソース・マネージャに対する保留中の変更を実行します。
<a href="#">SET_INITIAL_CONSUMER_GROUP プロシージャ</a> (39-16 ページ)	ユーザーに対して、初期リソース・コンシューマ・グループを割り当てます。
<a href="#">SWITCH_CONSUMER_GROUP_FOR_SESS プロシージャ</a> (39-17 ページ)	指定のセッションのリソース・コンシューマ・グループを変更します。
<a href="#">SWITCH_CONSUMER_GROUP_FOR_USER プロシージャ</a> (39-17 ページ)	指定のユーザー名で、すべてのセッションのリソース・コンシューマ・グループを変更します。

## CREATE\_PLAN プロシージャ

このプロシージャは、リソース・プランを定義するエントリを作成します。

### 構文

```
DBMS_RESOURCE_MANAGER.CREATE_PLAN (
    plan                               IN VARCHAR2,
    comment                           IN VARCHAR2,
    cpu_mth                           IN VARCHAR2 DEFAULT 'EMPHASIS',
    max_active_sess_target_mth        IN VARCHAR2 DEFAULT 'MAX_ACTIVE_SESS_ABSOLUTE',
    parallel_degree_limit_mth         IN VARCHAR2 DEFAULT 'PARALLEL_DEGREE_LIMIT_
ABSOLUTE');
```

### パラメータ

表 39-2 CREATE\_PLAN プロシージャのパラメータ

パラメータ	説明
plan	リソース・プラン名。
cpu_mth	CPU リソースに対する割当て方法。
max_active_sess_target_mth	アクティブな最大セッションに対する割当て方法。
parallel_degree_limit_mth	並列度に対する割当て方法。
comment	ユーザー・コメント。

## UPDATE\_PLAN プロシージャ

このプロシージャは、リソース・プランを定義するエントリを更新します。

### 構文

```
DBMS_RESOURCE_MANAGER.UPDATE_PLAN (  
    plan                                IN VARCHAR2,  
    new_comment                        IN VARCHAR2 DEFAULT NULL,  
    new_cpu_mth                       IN VARCHAR2 DEFAULT NULL,  
    new_max_active_sess_target_mth IN VARCHAR2 DEFAULT NULL,  
    new_parallel_degree_limit_mth IN VARCHAR2 DEFAULT NULL);
```

### パラメータ

表 39-3 UPDATE\_PLAN プロシージャのパラメータ

パラメータ	説明
plan	リソース・プラン名。
new_comment	新規のユーザー・コメント。
new_cpu_mth	CPU リソースに対する新規の割当て方法名。
new_max_active_sess_target_mth	アクティブな最大セッションに対する新規の方法名。
new_parallel_degree_limit_mth	並列度に対する新規の方法名。

### 使用上の注意

UPDATE\_PLAN に対するパラメータを指定しない場合、これらのパラメータは、データ・ディクショナリ内で変更されないまま残ります。

## DELETE\_PLAN プロシージャ

このプロシージャは、指定のプランとそれが参照するすべてのプラン・ディレクティブを削除します。

### 構文

```
DBMS_RESOURCE_MANAGER.DELETE_PLAN (  
    plan IN VARCHAR2);
```



## パラメータ

表 39-4 DELETE\_PLAN プロシージャのパラメータ

パラメータ	説明
plan	削除するリソース・プランの名前。

## DELETE\_PLAN\_CASCADE プロシージャ

このプロシージャは、指定のプランとそのすべての子（プラン・ディレクティブ、サブプラン、コンシューマ・グループ）を削除します。必須オブジェクトと必須ディレクティブは削除されません。

### 構文

```
DBMS_RESOURCE_MANAGER.DELETE_PLAN_CASCADE (  
    plan IN VARCHAR2);
```

## パラメータ

表 39-5 DELETE\_PLAN\_CASCADE プロシージャのパラメータ

パラメータ	説明
plan	プラン名。

### エラー

DELETE\_PLAN\_CASCADE プロシージャでエラーが発生した場合は、ロールバックされるため、何も削除されません。

**注意：** デフォルトのリソース割当て方法を使用する場合は、プランの作成または更新時に方法を指定する必要はありません。

### 使用上の注意

デフォルトは次のとおりです。

- cpu\_method = EMPHASIS
- parallel\_degree\_limit\_mth = PARALLEL\_DEGREE\_LIMIT\_ABSOLUTE
- max\_active\_sess\_target\_mth = MAX\_ACTIVE\_SESS\_ABSOLUTE

**注意：** パラメータ `max_active_sess_target_mth` は、このリリースでは記載されていません。このパラメータは、将来使用するために予約されています。

## CREATE\_CONSUMER\_GROUP プロシージャ

このプロシージャによって、リソース・コンシューマ・グループを定義するエントリを作成します。

### 構文

```
DBMS_RESOURCE_MANAGER.CREATE_CONSUMER_GROUP (
    consumer_group IN VARCHAR2,
    comment        IN VARCHAR2,
    cpu_mth        IN VARCHAR2 DEFAULT 'ROUND-ROBIN');
```

### パラメータ

表 39-6 CREATE\_CONSUMER\_GROUP プロシージャのパラメータ

パラメータ	説明
consumer_group	コンシューマ・グループ名。
comment	ユーザー・コメント。
cpu_mth	CPU リソース割当て方法名。

## UPDATE\_CONSUMER\_GROUP プロシージャ

このプロシージャによって、リソース・コンシューマ・グループを定義するエントリを更新します。

### 構文

```
DBMS_RESOURCE_MANAGER.UPDATE_CONSUMER_GROUP (
    consumer_group IN VARCHAR2,
    new_comment    IN VARCHAR2 DEFAULT NULL,
    new_cpu_mth    IN VARCHAR2 DEFAULT NULL);
```

## パラメータ

表 39-7 UPDATE\_CONSUMER\_GROUP プロシージャのパラメータ

パラメータ	説明
consumer_group	コンシューマ・グループ名。
new_comment	新規のユーザー・コメント。
new_cpu_mth	CPU リソース割当てに対する新規の方法名。

UPDATE\_CONSUMER\_GROUP に対するパラメータを指定しない場合、これらのパラメータは、データ・ディクショナリ内で変更されないまま残ります。

## DELETE\_CONSUMER\_GROUP プロシージャ

このプロシージャによって、リソース・コンシューマ・グループを定義するエントリを削除します。

### 構文

```
DBMS_RESOURCE_MANAGER.DELETE_CONSUMER_GROUP (  
    consumer_group IN VARCHAR2);
```

## パラメータ

表 39-8 DELETE\_CONSUMER\_GROUP プロシージャのパラメータ

パラメータ	説明
consumer_group	削除するコンシューマ・グループの名前。

## CREATE\_PLAN\_DIRECTIVE プロシージャ

このプロシージャによって、リソース・プラン・ディレクティブを作成します。

### 構文

```
DBMS_RESOURCE_MANAGER.CREATE_PLAN_DIRECTIVE (  
    plan                                IN VARCHAR2,  
    group_or_subplan                   IN VARCHAR2,  
    comment                             IN VARCHAR2,  
    cpu_p1                             IN NUMBER    DEFAULT NULL,  
    cpu_p2                             IN NUMBER    DEFAULT NULL,  
    cpu_p3                             IN NUMBER    DEFAULT NULL,  
    cpu_p4                             IN NUMBER    DEFAULT NULL,  
    cpu_p5                             IN NUMBER    DEFAULT NULL,  
    cpu_p6                             IN NUMBER    DEFAULT NULL,  
    cpu_p7                             IN NUMBER    DEFAULT NULL,  
    cpu_p8                             IN NUMBER    DEFAULT NULL,  
    max_active_sess_target_p1          IN NUMBER    DEFAULT NULL,  
    parallel_degree_limit_p1           IN NUMBER    DEFAULT NULL);
```

### パラメータ

表 39-9 CREATE\_PLAN\_DIRECTIVE プロシージャのパラメータ

パラメータ	説明
plan	リソース・プラン名。
group_or_subplan	コンシューマ・グループ名またはサブプラン名。
comment	プラン・ディレクティブについてのコメント。
cpu_p1	CPU リソース割当て方法に対する第 1 パラメータ。
cpu_p2	CPU リソース割当て方法に対する第 2 パラメータ。
cpu_p3	CPU リソース割当て方法に対する第 3 パラメータ。
cpu_p4	CPU リソース割当て方法に対する第 4 パラメータ。
cpu_p5	CPU リソース割当て方法に対する第 5 パラメータ。
cpu_p6	CPU リソース割当て方法に対する第 6 パラメータ。
cpu_p7	CPU リソース割当て方法に対する第 7 パラメータ。
cpu_p8	CPU リソース割当て方法に対する第 8 パラメータ。
max_active_sess_target_p1	最大アクティブ・セッション割当て方法に対する第 1 パラメータ (将来使用のために予約)。

表 39-9 CREATE\_PLAN\_DIRECTIVE プロシージャのパラメータ

パラメータ	説明
parallel_degree_limit_p1	並列度の割当て方法に対する第 1 パラメータ。

すべてのパラメータは、NULL にデフォルト設定されます。ただし、EMPHASIS CPU リソース割当て方法の場合は、ユーザーがすべてのパラメータを入力する必要があります。

## UPDATE\_PLAN\_DIRECTIVE プロシージャ

このプロシージャによって、リソース・プラン・ディレクティブを更新します。

### 構文

```
DBMS_RESOURCE_MANAGER.UPDATE_PLAN_DIRECTIVE (
    plan                               IN VARCHAR2,
    group_or_subplan                   IN VARCHAR2,
    new_comment                         IN VARCHAR2 DEFAULT NULL,
    new_cpu_p1                         IN NUMBER   DEFAULT NULL,
    new_cpu_p2                         IN NUMBER   DEFAULT NULL,
    new_cpu_p3                         IN NUMBER   DEFAULT NULL,
    new_cpu_p4                         IN NUMBER   DEFAULT NULL,
    new_cpu_p5                         IN NUMBER   DEFAULT NULL,
    new_cpu_p6                         IN NUMBER   DEFAULT NULL,
    new_cpu_p7                         IN NUMBER   DEFAULT NULL,
    new_cpu_p8                         IN NUMBER   DEFAULT NULL,
    new_max_active_sess_target_p1     IN NUMBER   DEFAULT NULL,
    new_parallel_degree_limit_p1      IN NUMBER   DEFAULT NULL);
```

### パラメータ

表 39-10 UPDATE\_PLAN\_DIRECTIVE プロシージャのパラメータ

パラメータ	説明
plan	リソース・プラン名。
group_or_subplan	コンシューマ・グループ名またはサブプラン名。
comment	プラン・ディレクティブについてのコメント。
cpu_p1	CPU リソース割当て方法に対する第 1 パラメータ。
cpu_p2	CPU リソース割当て方法に対する第 2 パラメータ。
cpu_p3	CPU リソース割当て方法に対する第 3 パラメータ。

表 39-10 UPDATE\_PLAN\_DIRECTIVE プロシージャのパラメータ

パラメータ	説明
cpu_p4	CPU リソース割当て方法に対する第 4 パラメータ。
cpu_p5	CPU リソース割当て方法に対する第 5 パラメータ。
cpu_p6	CPU リソース割当て方法に対する第 6 パラメータ。
cpu_p7	CPU リソース割当て方法に対する第 7 パラメータ。
cpu_p8	CPU リソース割当て方法に対する第 8 パラメータ。
max_active_sess_target_p1	最大アクティブ・セッション割当て方法に対する第 1 パラメータ (将来使用のために予約)。
parallel_degree_limit_p1	並列度の割当て方法に対する第 1 パラメータ。

UPDATE\_PLAN\_DIRECTIVE に対してパラメータを指定しない場合、これらのパラメータは、データ・ディクショナリ内で変更されないまま残ります。

DELETE\_PLAN\_DIRECTIVE プロシージャ

このプロシージャによって、リソース・プラン・ディレクティブを削除します。

構文

```
DBMS_RESOURCE_MANAGER.DELETE_PLAN_DIRECTIVE (  
    plan                IN VARCHAR2,  
    group_or_subplan IN VARCHAR2);
```

パラメータ

表 39-11 DELETE\_PLAN\_DIRECTIVE プロシージャのパラメータ

パラメータ	説明
plan	リソース・プラン名。
group_or_subplan	グループ名またはサブプラン名。

CREATE\_PENDING\_AREA プロシージャ

このプロシージャによって、リソース・マネージャ・オブジェクトに変更を加えます。

プラン・スキーマへのすべての変更は、保留領域内で行う必要があります。保留領域は、プラン・スキーマを変更するためのスクラッチ領域とみなすことができます。管理者は、この

保留領域を作成し、必要に応じて変更を加え、場合によってその変更を検証します。そして、その実行が完了したときのみ、その変更内容がアクティブになります。

保留領域がアクティブな間は、変更された現行のプラン・スキーマを適切なユーザー・ビューから選択して、いつでも表示できます。

現在の変更を中止する場合は、いつでも保留領域を消去できます。また、`VALIDATE` プロシージャをコールして、変更が有効になっているかどうかを確認できます。変更は、エントリ・グループの一貫性を維持するための指定の順序で行う必要はありません。これらのチェックは、保留領域が実行されるときにも暗黙的に行われます。

---

**注意：** Oracle では、親なしコンシューマ・グループ（つまり、そのコンシューマ・グループを参照するプラン・ディレクティブがないコンシューマ・グループ）が可能です。これは、現在は使用しないが将来使用するコンシューマ・グループを管理者があらかじめ作成できるようにするためです。

---

## 構文

```
DBMS_RESOURCE_MANAGER.CREATE_PENDING_AREA;
```

## パラメータ

なし。

## 使用上の注意

次のルールを厳守してください。これらのルールは、`VALIDATE` または `SUBMIT` プロシージャが実行されるたびにチェックされます。

1. プラン・スキーマにループがないこと。
2. プラン・ディレクティブが参照するすべてのプランとコンシューマ・グループがあること。
3. すべてのプランに、プランまたはコンシューマ・グループのいずれかを参照するプラン・ディレクティブがあること。
4. リソース割当て方法が `EMPHASIS` の場合は、指定レベルでのパーセントの合計が 100 を超えないこと。
5. アクティブなインスタンスでトップレベルのプランとして現在使用されているプランを削除しないこと。
6. Oracle8i の場合、プラン・ディレクティブのパラメータ `parallel_degree_limit_pl` は、コンシューマ・グループ（サブプランではなく）を参照するプラン・ディレクティブでのみ表示されます。

7. 指定のプランでのプラン・ディレクティブは 32 を超えないこと（つまり、プランは 32 を超える子を持つことはできません）。
8. アクティブなプラン・スキーマ内のコンシューマ・グループは 32 を超えないこと。
9. プランとコンシューマ・グループは同じ名前領域を使用するため、コンシューマ・グループと同じ名前のプランがないこと。
10. アクティブなプラン・スキーマ内に OTHER\_GROUPS に対するプラン・ディレクティブがあること。これにより、現在アクティブなプランがカバーしていないセッションに OTHER\_GROUPS ディレクティブが指定したリソースが割り当てられます。

VALIDATE または SUBMIT プロシージャによるチェック時に、前述のルールのいずれかに違反していると、それを通知するエラー・メッセージが戻されます。変更して問題を修正し、VALIDATE または SUBMIT プロシージャを再発行できます。

## VALIDATE\_PENDING\_AREA プロシージャ

このプロシージャによって、リソース・マネージャに対する保留中の変更内容を検証します。

### 構文

```
DBMS_RESOURCE_MANAGER.VALIDATE_PENDING_AREA;
```

### パラメータ

なし。

## CLEAR\_PENDING\_AREA プロシージャ

このプロシージャによって、リソース・マネージャに対する保留中の変更内容を消去します。

### 構文

```
DBMS_RESOURCE_MANAGER.CLEAR_PENDING_AREA;
```

### パラメータ

なし。

## SUBMIT\_PENDING\_AREA プロシージャ

このプロシージャによって、リソース・マネージャに対する保留中の変更を発行します。変更内容を検証してコミットした後（その変更内容が有効な場合）保留領域を消去します。



**注意:** SUBMIT\_PENDING\_AREA へのコールは、VALIDATE\_PENDING\_AREA が成功していても失敗する場合があります。これは、削除するプランがインスタンスによって VALIDATE\_PENDING\_AREA へのコール後、SUBMIT\_PENDING\_AREA へのコールまでにロードされると発生する可能性があります。

構文

```
DBMS_RESOURCE_MANAGER.SUBMIT_PENDING_AREA;
```

パラメータ

なし。

例

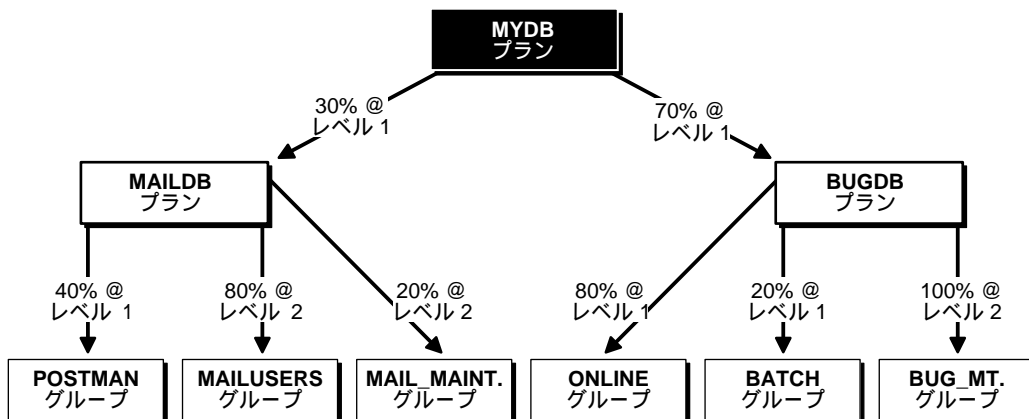
プランの利点の 1 つは、相互に参照できることです。プランのエントリは、コンシューマ・グループまたはサブプランのいずれかになります。次に、有効な CPU プラン・ディレクティブのセット例を示します。

表 39-12 MYDB PLAN CPU プラン・ディレクティブ

サブプラン / グループ	CPU_Level 1
MAILDB プラン	30%
BUGDB プラン	70%

これらのプラン・ディレクティブが有効で、すべてのコンシューマ・グループに実行可能なセッションが無限にある場合、MAILDB プランには使用可能な CPU リソースの 30% が割り当てられ、一方、BUGDB プランにはその 70% が割り当てられます。これをさらに分割すると、"Postman" コンシューマ・グループにあるセッションは 12% ( 30% の内の 40% ) の時間を実行し、"Online" コンシューマ・グループにあるセッションは 56% ( 70% の内の 80% ) の時間を実行します。

次の図は、このシナリオを表しています。



次の説明では、コンシューマ・グループはアクティブなセッションです。つまり、セッションはリソース・コンシューマ・グループに所属し、このコンシューマ・グループは、処理するリソースの割当てを決定するためにプランが使用します。

CPU プラン・ディレクティブのマルチプラン（1 つ以上のサブプランをもつプラン）定義は、各プランが自分自身をエンティティとして所有するため、1 セットのプラン・ディレクティブを持つ単一プランに縮小できません。プランまたはサブプランに割り当てられた CPU 量は、アクティブ・セッションを持つコンシューマ・グループがそのプランに含まれていない限り、そのプラン内でのみ使用されます。したがって、この例では、Bug Maintenance グループが CPU 量をまったく使用しなかった場合はそのプラン内でリサイクルされるので、BUGDB プラン内のレベル 1 に戻ります。前述の例で、マルチプラン定義が複数のコンシューマ・グループを持つ単一プランに縮小された場合は、Bug Maintenance Group で使用されていない CPU 量を明示的にリサイクルする方法はありません。CPU 量はグローバルにリサイクルされるので、MAIL セッションにもそれを使用する機会が与えられます。

データベースのリソースは、複数のアプリケーション間の高いレベルでパーティション化でき、アプリケーション内で再パーティション化できます。アプリケーション内の指定のグループで、割り当てられたすべてのリソースが必要ない場合、そのリソースは、同じアプリケーション内でのみ再パーティション化されます。

次の例では、プランとコンシューマ・グループのデフォルトの割当て方法が使用されます。

```

create_pending_area();
create_plan(plan => 'BUGDB_PLAN', comment => 'Resource
  plan/method for bug users' sessions');
create_plan(plan => 'MAILDB_PLAN', comment => 'Resource
  plan/method for mail users' sessions');
create_plan(plan => 'MYDB_PLAN', comment => 'Resource

```

```
plan/method for bug and mail users' sessions');
create_consumer_group(consumer_group => 'Bug_Online_group',
    comment => 'Resource consumer group/method for online bug users'
    sessions');
create_consumer_group(consumer_group => 'Bug_Batch_group',
    comment => 'Resource consumer group/method for bug users' sessions
    who run batch jobs');
create_consumer_group(consumer_group =>
    'Bug_Maintenance_group', comment => 'Resource consumer
    group/method for users' sessions who maintain the bug db');
create_consumer_group(consumer_group => 'Mail_users_group',
    comment => 'Resource consumer group/method for mail users'
    sessions');
create_consumer_group(consumer_group => 'Mail_Postman_group',
    comment => 'Resource consumer group/method for mail postman');
create_consumer_group(consumer_group =>
    'Mail_Maintenance_group', comment => 'Resource consumer
    group/method for users' sessions who maintain the mail db');
create_plan_directive(plan => 'BUGDB_PLAN', group_or_subplan
    => 'Bug_Online_group', comment => 'online bug users'
    sessions at level 0', cpu_p1 => 80, cpu_p2=> 0,
    parallel_degree_limit_p1 => 8);
create_plan_directive(plan => 'BUGDB_PLAN', group_or_subplan
    => 'Bug_Batch_group', comment => 'batch bug users'
    sessions at level 0', cpu_p1 => 20, cpu_p2 => 0,
    parallel_degree_limit_p1 => 2);
create_plan_directive(plan => 'BUGDB_PLAN', group_or_subplan
    => 'Bug_Maintenance_group', comment => 'bug maintenance users'
    sessions at level 1', cpu_p1 => 0, cpu_p2 => 100,
    parallel_degree_limit_p1 => 3);
create_plan_directive(plan => 'MAILDB_PLAN', group_or_subplan
    => 'Mail_Postman_group', comment => 'mail postman at
    level 0', cpu_p1 => 40, cpu_p2 => 0,
    parallel_degree_limit_p1 => 4);
create_plan_directive(plan => 'MAILDB_PLAN', group_or_subplan
    => 'Mail_users_group', comment => 'mail users' sessions
    at level 1', cpu_p1 => 0, cpu_p2 => 80,
    parallel_degree_limit_p1 => 4);
create_plan_directive(plan => 'MAILDB_PLAN', group_or_subplan =>
    'Mail_Maintenance_group', comment => 'mail
    maintenance users' sessions at level 1', cpu_p1 => 0,
    cpu_p2 => 20, parallel_degree_limit_p1 => 2);
create_plan_directive(plan => 'MYDB_PLAN', group_or_subplan =>
    'MAILDB_PLAN', comment=> 'all mail users' sessions at
    level 0', cpu_p1 => 30);
create_plan_directive(plan => 'MYDB_PLAN', group_or_subplan =>
    'BUGDB_PLAN', comment => 'all bug users' sessions at
```

```
level 0', cpu_pl = 70);  
validate_pending_area();  
submit_pending_area();
```

検証は SUBMIT\_PENDING\_AREA で暗黙的に行われるので、前述の VALIDATE\_PENDING\_AREA へのコールはオプションです。

SET\_INITIAL\_CONSUMER\_GROUP プロシージャ

ユーザーの初期コンシューマ・グループは、そのユーザーが作成したセッションが最初に所属しているコンシューマ・グループです。このプロシージャは、ユーザーに対して、初期のリソース・コンシューマ・グループを設定します。

構文

```
DBMS_RESOURCE_MANAGER.SET_INITIAL_CONSUMER_GROUP (  
    user          IN VARCHAR2,  
    consumer_group IN VARCHAR2);
```

パラメータ

表 39-13 SET\_INITIAL\_CONSUMER\_GROUP プロシージャのパラメータ

パラメータ	説明
user	ユーザー名。
consumer_group	ユーザーの初期コンシューマ・グループ。

使用上の注意

このプロシージャを実行するためには、ADMINISTER\_RESOURCE\_MANAGER または ALTER USER システム権限が必要です。ユーザーの初期コンシューマ・グループが設定される前に、ユーザーまたは PUBLIC に対して、コンシューマ・グループへのスイッチ権限が直接付与されている必要があります。初期コンシューマ・グループに対するスイッチ権限は、そのユーザーに付与されているロールから与えることはできません。

**注意：** この方法は、ALTER USER DEFAULT ROLE に対する方法に類似しています。

ユーザーの初期コンシューマ・グループが設定されていない場合、そのユーザーの初期コンシューマ・グループは、自動的にコンシューマ・グループ DEFAULT\_CONSUMER\_GROUP になります。

DEFAULT\_CONSUMER\_GROUP は PUBLIC に付与されたスイッチ権限を持つため、すべてのユーザーはこのコンシューマ・グループに対するスイッチ権限を自動的に付与されます。コンシューマ・グループを削除するとき、削除するグループを初期コンシューマ・グループとしていたすべてのユーザーは、DEFAULT\_CONSUMER\_GROUP を初期コンシューマ・グループとします。削除するコンシューマ・グループに所属している現行のアクティブなセッションは、すべて DEFAULT\_CONSUMER\_GROUP に切り替えられます。

## SWITCH\_CONSUMER\_GROUP\_FOR\_SESS プロシージャ

このプロシージャによって、指定のセッションのリソース・コンシューマ・グループを変更します。また、トップレベルのユーザー・セッションに関連する (PQ) スレーブ・セッションのコンシューマ・グループも変更します。

### 構文

```
DBMS_RESOURCE_MANAGER.SWITCH_CONSUMER_GROUP_FOR_SESS (
    session_id      IN NUMBER,
    session_serial  IN NUMBER,
    consumer_group  IN VARCHAR2);
```

### パラメータ

表 39-14 SWITCH\_CONSUMER\_GROUP\_FOR\_SESS プロシージャのパラメータ

パラメータ	説明
session_id	ビュー v\$SESSION での SID 列。
session_serial	ビュー v\$SESSION での SERIAL# 列。
consumer_group	切り替えるコンシューマ・グループの名前。

## SWITCH\_CONSUMER\_GROUP\_FOR\_USER プロシージャ

このプロシージャによって、指定のユーザー ID を持つすべてのセッションに対するリソース・コンシューマ・グループを変更します。また、トップレベルのユーザー・セッションに関連する (PQ) スレーブ・セッションのコンシューマ・グループも変更します。

### 構文

```
DBMS_RESOURCE_MANAGER.SWITCH_CONSUMER_GROUP_FOR_USER (
    user            IN VARCHAR2,
    consumer_group  IN VARCHAR2);
```

パラメータ

表 39-15 SWITCH\_CONSUMER\_GROUP\_FOR\_USER プロシージャのパラメータ

パラメータ	説明
user	ユーザー名。
consumer_group	切り替えるコンシューマ・グループの名前。

使用上の注意

SWITCH\_CONSUMER\_GROUP\_FOR\_SESS プロシージャと SWITCH\_CONSUMER\_GROUP\_FOR\_USER プロシージャによって、特定のセッションまたはユーザーの CPU リソース割当てを増減します。これにより、UNIX の nice コマンドと類似した機能性が提供されます。

これらのプロシージャは、新規に指定されたコンシューマ・グループにセッションを即時に移動します。

---

## DBMS\_RESOURCE\_MANAGER\_PRIVS

DBMS\_RESOURCE\_MANAGER\_PRIVS パッケージは、リソース・マネージャに関連付けられている権限をメンテナンスします。

**関連項目：** データベース・リソース・マネージャの使用方法的詳細は、『Oracle8i 管理者ガイド』を参照してください。

# サブプログラムの要約

表 40-1 DBMS\_RESOURCE\_MANAGER\_PRIVS パッケージのサブプログラム

サブプログラム	説明
<a href="#">GRANT_SYSTEM_PRIVILEGE プロシージャ</a> (40-2 ページ)	システム権限を付与します。
<a href="#">REVOKE_SYSTEM_PRIVILEGE プロシージャ</a> (40-3 ページ)	システム権限を取り消します。
<a href="#">GRANT_SWITCH_CONSUMER_GROUP プロシージャ</a> (40-3 ページ)	リソース・コンシューマ・グループに切り替える権限を付与します。
<a href="#">REVOKE_SWITCH_CONSUMER_GROUP プロシージャ</a> (40-5 ページ)	リソース・コンシューマ・グループに切り替える権限を取り消します。

## GRANT\_SYSTEM\_PRIVILEGE プロシージャ

このプロシージャは、ユーザーまたはロールにシステム権限を付与します。

### 構文

```
DBMS_RESOURCE_MANAGER_PRIVS.GRANT_SYSTEM_PRIVILEGE (  
    grantee_name    IN VARCHAR2,  
    privilege_name  IN VARCHAR2 DEFAULT 'ADMINISTER_RESOURCE_MANAGER',  
    admin_option    IN BOOLEAN);
```

### パラメータ

表 40-2 GRANT\_SYSTEM\_PRIVILEGE プロシージャのパラメータ

パラメータ	説明
grantee_name	権限が付与されるユーザーまたはロールの名前。
privilege_name	付与される権限の名前。
admin_option	admin_option の付いた権限付与の場合は TRUE、そうでない場合は FALSE。

Oracle では現在、リソース・マネージャに対するシステム権限は ADMINISTER\_RESOURCE\_MANAGER のみ提供しています。データベース管理者には、admin option を伴うこのシステム権限があります。その権限の付与と取消しは、ユーザーまたはロールに対して行うことができます。admin option を伴うシステム権限を付与されたユーザーは、この権限を他のユーザーにも付与できます。



## 例

次のコールは、scott というユーザーに対して admin option を付けないでこの権限を付与します。

```
DBMS_RESOURCE_MANAGER_PRIVS.GRANT_SYSTEM_PRIVILEGE (  
    grantee_name => 'scott',  
    admin_option => FALSE);
```

## REVOKE\_SYSTEM\_PRIVILEGE プロシージャ

このプロシージャは、ユーザーまたはロールからシステム権限を取り消します。

## 構文

```
DBMS_RESOURCE_MANAGER_PRIVS.REVOKE_SYSTEM_PRIVILEGE (  
    revokee_name    IN VARCHAR2,  
    privilege_name  IN VARCHAR2 DEFAULT 'ADMINISTER_RESOURCE_MANAGER');
```

## パラメータ

表 40-3 REVOKE\_SYSTEM\_PRIVILEGE プロシージャのパラメータ

パラメータ	説明
revokee_name	権限を取り消すユーザーまたはロールの名前。
privilege_name	取り消す権限の名前。

## 例

次のコールは、ユーザー名 scott から ADMINISTER\_RESOURCE\_MANAGER を取り消します。

```
DBMS_RESOURCE_MANAGER_PRIVS.REVOKE_SYSTEM_PRIVILEGE ('scott');
```

## GRANT\_SWITCH\_CONSUMER\_GROUP プロシージャ

このプロシージャは、リソース・コンシューマ・グループに切り替える権限を付与します。

## 構文

```
DBMS_RESOURCE_MANAGER_PRIVS.GRANT_SWITCH_CONSUMER_GROUP (  
    grantee_name    IN VARCHAR2,  
    consumer_group  IN VARCHAR2,  
    grant_option    IN BOOLEAN);
```

パラメータ

表 40-4 GRANT\_SWITCH\_CONSUMER\_GROUP プロシージャのパラメータ

パラメータ	説明
grantee_name	権限を付与されるユーザーまたはロールの名前。
consumer_group	コンシューマ・グループ名。
grant_option	権限を付与されたユーザーが他のユーザーにアクセス権を付与できる場合は TRUE、そうでない場合は FALSE。

使用上の注意

ユーザーに対して特定のコンシューマ・グループに切り替える許可を付与すると、そのユーザーは、即時に現行のコンシューマ・グループを新規のコンシューマ・グループに切り替えることができます。

ロールに対して特定のコンシューマ・グループに切り替える許可を付与すると、そのロールを付与され、そのロールを使用可能にしたユーザーは、即時に現行のコンシューマ・グループを新規のコンシューマ・グループに切り替えることができます。

PUBLIC に対して特定のコンシューマ・グループに切り替える許可を付与すると、すべてのユーザーがそのコンシューマ・グループに切り替えることができます。

grant\_option パラメータが TRUE の場合、コンシューマ・グループに対するスイッチ権限が付与されたユーザーは、そのコンシューマ・グループに対するスイッチ権限を他のユーザーにも付与できます。

ユーザーの初期のコンシューマ・グループを設定するには、そのグループに対するスイッチ権限をユーザーに付与する必要があります。

関連項目： [第 39 章「DBMS\\_RESOURCE\\_MANAGER」](#)

例

```
DBMS_RESOURCE_MANAGER.PRIVS.GRANT_SWITCH_CONSUMER_GROUP (  
    'scott', 'mail_maintenance_group', true);  
  
DBMS_RESOURCE_MANAGER.SET_INITIAL_CONSUMER_GROUP (  
    'scott', 'mail_maintenance_group');
```

## REVOKE\_SWITCH\_CONSUMER\_GROUP プロシージャ

このプロシージャは、リソース・コンシューマ・グループに切り替える権限を取り消します。

### 構文

```
DBMS_RESOURCE_MANAGER_PRIVS.REVOKE_SWITCH_CONSUMER_GROUP (  
    revokee_name    IN VARCHAR2,  
    consumer_group  IN VARCHAR2);
```

### パラメータ

表 40-5 REVOKE\_SWITCH\_CONSUMER\_GROUP プロシージャのパラメータ

パラメータ	説明
revokee_name	アクセス権を取り消すユーザーまたはロールの名前。
consumer_group	コンシューマ・グループ名。

### 使用上の注意

特定のコンシューマ・グループに対するスイッチ権限をユーザーから取り消した後で、そのユーザーがそのコンシューマ・グループに切り替えようとしても失敗します。

ユーザーから初期のコンシューマ・グループを取り消した場合、そのユーザーは、ログイン時に自動的に DEFAULT\_CONSUMER\_GROUP コンシューマ・グループに所属します。

コンシューマ・グループに対するスイッチ権限をロールから取り消すと、そのロールを介してのみコンシューマ・グループに対するスイッチ権限を持っていたユーザーは、以降そのコンシューマ・グループに切り替えることはできません。

コンシューマ・グループに対するスイッチ権限を PUBLIC から取り消すと、それまで PUBLIC を介してのみコンシューマ・グループを使用できたユーザーは、以降そのコンシューマ・グループに切り替えることはできません。

### 例

次の例は、mail\_maintenance\_group に切り替える権限を scott から取り消します。

```
DBMS_RESOURCE_MANAGER_PRIVS.REVOKE_SWITCH_CONSUMER_GROUP (  
    'scott', 'mail_maintenance_group');
```



DBMS\_RLS パッケージには、ファイン・グレイン・アクセス・コントロールの管理インタフェースが含まれています。

---

**注意：** DBMS\_RLS は、Enterprise Edition でのみ使用できます。

---

---

## 動的な述語

ファイン・グレイン・アクセス・コントロールをサポートする機能は、動的な述語に基づいています。セキュリティ・ルールはビューに埋め込まれていませんが、実表またはビューが DML 文で参照される、文の解析時に取得されます。

表またはビューに対する動的な述語は PL/SQL ファンクションが生成し、PL/SQL インタフェースを介してセキュリティ・ポリシーに関連付けられます。次に例を示します。

```
DBMS_RLS.ADD_POLICY (  
    'scott', 'emp', 'emp_policy', 'secusr', 'emp_sec', 'select');
```

SCOTT スキーマの下にある EMP 表が問合せまたは副問合せ (SELECT) で参照されるたびに、サーバーは、EMP\_SEC ファンクション (SECUSR スキーマの下にある) をコールします。このファンクションは、EMP\_POLICY ポリシーについて、現ユーザーに固有の述語を戻します。ポリシー・ファンクションは、ファンクション・コール時に使用可能なすべてのセッション環境変数に基づいて述語を生成できます。これらの変数は通常、アプリケーション・コンテキストのフォームで表示されます。

次に、サーバーは、テキストがある一時ビューを作成します。

```
SELECT * FROM scott.emp WHERE P1
```

ここで、P1 (SAL > 10000 や副問合せなど) は、EMP\_SEC ファンクションから戻された述語です。サーバーは、EMP 表をビューとして処理し、ビューのテキストをデータ・ディクショナリからではなく一時ビューから取得すること以外は、通常のビューと同様にビューの展開を行います。

述語に副問合せがある場合は、ポリシー・ファンクションの所有者 (定義者) を使用して副問合せ内のオブジェクトを解決し、そのオブジェクトのセキュリティをチェックします。つまり、ポリシー保護されたオブジェクトへのアクセス権限を持つユーザーには、ポリシーについての知識は不要です。このユーザーには、基礎となるセキュリティ・ポリシーに対するオブジェクト権限の付与は不要です。さらに、サーバーはファンクション定義者の権限でコールを行うため、このユーザーにはポリシー・ファンクションでの EXECUTE 権限は不要です。

---

**注意：** 一時ビューは、単一表または述語だけを持つビュー (つまり、JOIN、ORDER BY、GROUP BY などが無い) から導出されるため、親オブジェクトの更新可能性を保持することができます。

---

DBMS\_RLS パッケージは、セキュリティ・ポリシーを削除したり、使用可能または使用禁止にするためのインタフェースも提供します。たとえば、次の PL/SQL 文を使用して、EMP\_POLICY を削除または使用禁止にできます。

```
DBMS_RLS.DROP_POLICY('scott', 'emp', 'emp_policy');  
DBMS_RLS.ENABLE_POLICY('scott', 'emp', 'emp_policy', FALSE)
```

## セキュリティ

一時ビューが副問合せを使用して作成されると、セキュリティ・チェックが実行されます。ポリシー・ファンクションを持ち、動的な述語を生成するスキーマは、セキュリティ・チェックとオブジェクト検索のために、一時ビューの定義者となります。

## 使用上の注意

DBMS\_RLS プロシージャは、現行の DML トランザクションがある場合は、操作前にコミットします。ただし、プロシージャが DDL イベント・トリガーの内部にある場合、プロシージャは最初にコミットを実行しません。DDL トランザクションに関して、DBMS\_RLS は、DDL トランザクションの一部です。

たとえば、ユーザーは CREATE TABLE のトリガーを作成できます。トリガー内部で、ALTER TABLE を介して列を追加でき、DBMS\_RLS を介してポリシーを追加できます。これらすべての操作は、それぞれが DDL 文であっても、CREATE TABLE と同じトランザクション内にあります。CREATE TABLE は、トリガーが正常終了した場合のみ成功します。

## サブプログラムの要約

表 41-1 DBMS\_RLS のサブプログラム

サブプログラム	説明
<a href="#">ADD_POLICY プロシージャ</a> (41-3 ページ)	ファイン・グ레인・アクセス・コントロールのポリシーを表またはビューに作成します。
<a href="#">DROP_POLICY プロシージャ</a> (41-5 ページ)	ファイン・グ레인・アクセス・コントロールのポリシーを表またはビューから削除します。
<a href="#">REFRESH_POLICY プロシージャ</a> (41-6 ページ)	ポリシーに関連付けられているすべてのキャッシュ済みの文を再解析します。
<a href="#">ENABLE_POLICY プロシージャ</a> (41-7 ページ)	ファイン・グ레인・アクセス・コントロールのポリシーを使用可能または使用禁止にします。

## ADD\_POLICY プロシージャ

このプロシージャは、ファイン・グ레인・アクセス・コントロールのポリシーを表またはビューに作成します。

トランザクションがある場合、その現行トランザクションはこのプロシージャによって操作の実行前にコミットを実行します。ただし、そのトランザクションが DDL イベント・トリガー内にある場合は、最初にコミットを実行しません。

関連項目：「[使用上の注意](#)」( 41-3 ページ )

コミットは、操作の最後にも実行されます。

構文

```
DBMS_RLS.ADD_POLICY (  
    object_schema    IN VARCHAR2 := NULL,  
    object_name      IN VARCHAR2,  
    policy_name      IN VARCHAR2,  
    function_schema  IN VARCHAR2 := NULL,  
    policy_function  IN VARCHAR2,  
    statement_types  IN VARCHAR2 := NULL,  
    update_check     IN BOOLEAN  := FALSE,  
    enable           IN BOOLEAN  := TRUE);
```

パラメータ

表 41-2 ADD\_POLICY プロシージャのパラメータ

パラメータ	説明
object_schema	表またはビューを含んでいるスキーマ ( NULL の場合はログオン・ユーザー )。
object_name	ポリシーを追加する表またはビューの名前。
policy_name	追加するポリシーの名前。この名前は、表またはビュー内に対して一意でなければなりません。
function_schema	ポリシー・ファンクションのスキーマ ( NULL の場合はログオン・ユーザー )。
policy_function	ポリシーの述語を生成するファンクションの名前。ファンクションがパッケージ内で定義されている場合、パッケージ名は必ず存在する必要があります。
statement_types	ポリシーを適用する文タイプ SELECT、INSERT、UPDATE および DELETE を任意に組み合わせることができます。デフォルトでは、すべてのタイプが適用されます。
update_check	文タイプ INSERT または UPDATE に対するオプションの引数。デフォルトは FALSE です。update_check を TRUE に設定すると、サーバーは、挿入または更新後の値に対してもポリシーをチェックします。
enable	ポリシーの追加時に、そのポリシーを使用可能にするかどうかを示します。デフォルトは TRUE です。



## 使用上の注意

- SYS は、セキュリティ・ポリシーの制約を受けません。
- 動的な述語を生成するポリシー・ファンクションは、サーバーがコールします。次の例は、ファンクションのインタフェースを示します。

```
FUNCTION policy_function (object_schema IN VARCHAR2, object_name VARCHAR2)
    RETURN VARCHAR2
--- object_schema is the schema owning the table of view.
--- object_name is the name of table of view that the policy will apply.
```

ポリシー・ファンクションが戻す述語の最大長は、2,000 バイトです。

- ポリシー・ファンクションには、WNDS（データベースへの書き込み禁止状態）の純粹さレベルが必要です。

**関連項目：** RESTRICT\_REFERENCES プラグマの詳細は、『PL/SQL ユーザーズ・ガイドおよびリファレンス』を参照してください。

- 同じオブジェクトにある複数のポリシーから生成された動的な述語は、全述語の論理積（AND 条件）の結合効果があります。
- セキュリティ・チェックとオブジェクト検索は、動的な述語の副問合せで、オブジェクトのポリシー・ファンクションの所有者に対して実行されます。
- ファンクションが長さ 0（ゼロ）の述語を戻す場合は、ポリシーについて現ユーザーに適用する制限はないと解釈されます。
- 述語で表の別名が必要な場合（たとえば、親オブジェクトがタイプ表の場合は、表またはビューの名前自体を別名として使用する必要があります。サーバーは、一時ビューを "select c1, c2, ... from tab where <predicate>" のように構成します。
- ファンクションの妥当性チェックは、インストレーションを容易にしたり、インポートまたはエクスポート時に発生するその他の依存する問題を軽減するため、実行時に行われます。

## DROP\_POLICY プロシージャ

このプロシージャは、ファイン・グレイン・アクセス・コントロールのポリシーを表またはビューから削除します。

トランザクションがある場合、その現行トランザクションはこのプロシージャによって操作の実行前にコミットを実行します。ただし、そのトランザクションが DDL イベント・トリガー内にある場合は、最初にコミットを実行しません。

**関連項目：** 「[使用上の注意](#)」( 41-3 ページ )

コミットは、操作の最後にも実行されます。

構文

```
DBMS_RLS.DROP_POLICY (  
    object_schema IN VARCHAR2 := NULL,  
    object_name   IN VARCHAR2,  
    policy_name   IN VARCHAR2);
```

パラメータ

表 41-3 DROP\_POLICY プロシージャのパラメータ

パラメータ	説明
object_schema	表またはビューを含んでいるスキーマ ( NULL の場合はログオン・ユーザー )。
object_name	表またはビューの名前。
policy_name	表またはビューから削除するポリシーの名前。

REFRESH\_POLICY プロシージャ

このプロシージャは、ポリシーに関連付けられたすべてのキャッシュ済みの文を再解析します。これにより、ポリシーへの最新の変更内容がプロシージャの実行直後に有効になります。

トランザクションがある場合、その現行トランザクションはこのプロシージャによって操作の実行前にコミットを実行します。ただし、トランザクションが DDL イベント・トリガー内にある場合は、最初にコミットを実行しません。

関連項目：「[使用上の注意](#)」( 41-3 ページ )

コミットは、操作の最後にも実行されます。

構文

```
DBMS_RLS.REFRESH_POLICY (  
    object_schema IN VARCHAR2 := NULL,  
    object_name   IN VARCHAR2 := NULL,  
    policy_name   IN VARCHAR2 := NULL);
```

パラメータ

表 41-4 REFRESH\_POLICY プロシージャのパラメータ

パラメータ	説明
object_schema	表またはビューを含んでいるスキーマ。

表 41-4 REFRESH\_POLICY プロシージャのパラメータ

パラメータ	説明
object_name	ポリシーが関連付けられている表またはビューの名前。
policy_name	リフレッシュするポリシーの名前。

## エラー

使用禁止になっているポリシーをリフレッシュしようとする、プロシージャはエラーを戻します。

## ENABLE\_POLICY プロシージャ

このプロシージャは、ファイナリティ・グレイナリティ・アクセス・コントロールのポリシーを使用可能または使用禁止にします。ポリシーは、その作成時に使用可能になっています。

トランザクションがある場合、その現行トランザクションはプロシージャによって操作の実行前にコミットを実行します。ただし、トランザクションが DDL イベント・トリガー内にある場合は、最初にコミットを実行しません。

**関連項目：**「[使用上の注意](#)」( 41-3 ページ )

コミットは、操作の最後にも実行されます。

## 構文

```
DBMS_RLS.ENABLE_POLICY (  
    object_schema IN VARCHAR2 := NULL,  
    object_name   IN VARCHAR2,  
    policy_name   IN VARCHAR2,  
    enable        IN BOOLEAN);
```

## パラメータ

表 41-5 ENABLE\_POLICY プロシージャのパラメータ

パラメータ	説明
object_schema	表またはビューを含んでいるスキーマ ( NULL の場合はログオン・ユーザー )。
object_name	ポリシーが関連付けられている表またはビューの名前。
policy_name	使用可能または使用禁止にするポリシーの名前。

表 41-5 ENABLE\_POLICY プロシージャのパラメータ

パラメータ	説明
enable	ポリシーを使用可能にする場合は TRUE、使用禁止にする場合は FALSE。

例

次の例は、ファイン・グレイン・アクセス・コントロールのポリシーを実施するために必要なステップを示します。

Oracle HR アプリケーションで、PER\_PEOPLE は PER\_ALL\_PEOPLE 表に対するビューで、両方のオブジェクトとも APPS スキーマの下にあります。

```
CREATE TABLE per_all_people
    (person_id NUMBER(15),
     last_name VARCHAR2(30),
     emp_no VARCHAR2(15), ...);
CREATE VIEW per_people AS
    SELECT * FROM per_all_people;
```

社内のユーザー・ロールに基づいて、PER\_PEOPLE ビューへのアクセスを制限するセキュリティ・ポリシーが必要です。ポリシーの述語は、HR\_SECURITY パッケージにある SECURE\_PERSON ファンクションで生成できます。このパッケージは、スキーマ APPS の下にあり、HR アプリケーションに関連するすべてのセキュリティ・ポリシーをサポートするファンクションが含まれています。また、すべてのアプリケーション・コンテキストは、APPS\_SEC 名前領域の下にあります。

```
CREATE PACKAGE BODY hr_security IS
    FUNCTION secure_person(obj_schema VARCHAR2, obj_name VARCHAR2)
        RETURN VARCHAR2 IS
        d_predicate VARCHAR2(2000);
    BEGIN
        -- for users with HR_ROLE set to EMP, map logon user name
        -- to employee id. FND_USER table stores relationship
        -- among database users, application users,
        -- and people held in the HR person table.
        IF SYS_CONTEXT('apps_sec', 'hr_role') = 'EMP' THEN
            d_predicate = 'person_id IN
                (SELECT employee_id FROM apps.fnd_user
                 WHERE user_name = SYS_CONTEXT(''userenv'', ''session_
user''))';
            -- for users with HR_ROLE set to MGR (manager), map
            -- security profile id to a list of employee id that
            -- the user can access
        ELSE IF SYS_CONTEXT('apps_sec', 'hr_role') = 'MGR' THEN
            d_predicate = 'person_id IN
```

```

        (SELECT ppl.employee_id FROM per_person_list ppl WHERE
           ppl.security_profile_id = SYS_CONTEXT('apps_sec',
           'security_profile_id'))
        OR EXISTS (SELECT NULL FROM apps.per security_profiles psp
WHERE
           SYS_CONTEXT('apps_sec', 'security_profile_id') =
           psp.security_profile_id AND psp.view_all_flag = 'Y')));

ELSE
    d_predicate = '1=2'; -- deny access to other users, may use something
like 'keycol=null'
END IF;
RETURN d_predicate;
END secure_person;
END hr_security;

```

次のステップでは、PER\_PEOPLE ビューのポリシー（ここでは PER\_PEOPLE\_SEC）を、動的な述語を生成する HR\_SECURITY.SECURE\_PERSON ファンクションに関連付けます。

```

DBMS_RLS.ADD_POLICY('apps', 'per_people', 'per_people_sec', 'apps'
                    'hr_security.secure_person', 'select, update, delete');

```

ここで、PER\_PEOPLE ビューを含んだ SELECT、UPDATE および DELETE 文は、アプリケーション・コンテキスト HR\_ROLE の値に基づいて 3 つの述語から 1 つを取得します。

---

**注意：** PER\_ALL\_PEOPLE 表を保護するセキュリティ・ファンクションと同じファンクションを使用して、PER\_ADDRESSES 表を保護する動的な述語も生成できます。これは、どちらの表も、データへのアクセスを制限するポリシーが同じためです。

---



---

## DBMS\_ROWID

DBMS\_ROWID パッケージによって、ユーザーは ROWID を作成し、PL/SQL プログラムと SQL 文から ROWID に関する情報を取得します。ベース 64 文字の外部 ROWID を解釈するコードを書かないで、ROWID のデータ・ブロック番号、オブジェクト番号およびその他のコンポーネントを検索できます。

---

**注意：** DBMS\_ROWID は、ユニバーサルな ROWID （UROWID）とともに使用しません。

---

---

## 使用上の注意

このパッケージにあるファンクションの一部は、パラメータ ROWID だけを必要とします。このパラメータは、1 つの文字または PL/SQL ROWID で、その内容は必要に応じて制限または拡張されます。

DBMS\_ROWID のファンクションとプロシージャは PL/SQL コードからコールでき、そのファンクションは SQL 文で使用することもできます。

---

**注意：** ROWID\_INFO は 1 つのプロシージャです。このプロシージャは、PL/SQL コードでのみ使用できます。

---

DBMS\_ROWID パッケージからのファンクションは、他の組み込み式の SQL ファンクションと同じように使用できます。つまり、これらのファンクションは、式が使用できればどこでも使用できます。次の例では、EMP 表にある単一行のブロック番号だけを戻すために、ROWID\_BLOCK\_NUMBER が使用されています。

```
SELECT dbms_rowid.rowid_block_number(rowid)
       FROM emp
       WHERE ename = 'KING';
```

**PL/SQL の例** 次の例では、EMP 表にある行の ROWID を戻し、DBMS\_ROWID パッケージの ROWID\_OBJECT ファンクションを使用して、その ROWID からデータ・オブジェクト番号を抽出して表示します。

```
DECLARE
    object_no    INTEGER;
    row_id       ROWID;
    ...
BEGIN
    SELECT ROWID INTO row_id FROM emp
           WHERE empno = 7499;
    object_no := dbms_rowid.rowid_object(row_id);
    dbms_output.put_line('The obj. # is ' || object_no);
    ...
```

## 要件

このパッケージは、パッケージ所有者 (sys) ではなく、コール・ユーザーの権限で実行されます。



---

## ROWID のタイプ

RESTRICTED	制限 ROWID
EXTENDED	拡張 ROWID

次に例を示します。

```
rowid_type_restricted constant integer := 0;  
rowid_type_extended   constant integer := 1;
```

---

---

**注意：** 拡張 ROWID は、Oracle8i 以降でのみ使用されます。

---

---

## ROWID の検証結果

VALID	有効 ROWID
INVALID	無効 ROWID

次に例を示します。

```
rowid_is_valid   constant integer := 0;  
rowid_is_invalid constant integer := 1;
```

## オブジェクト型

UNDEFINED	(制限 ROWIDs に対する) オブジェクト番号が定義されていません。
-----------	--------------------------------------

次に例を示します。

```
rowid_object_undefined constant integer := 0;
```

## ROWID の変換タイプ

INTERNAL	ROWID タイプの列から、またはその列への変換。
EXTERNAL	文字列形式から、または文字列形式への変換。

次に例を示します。

```
rowid_convert_internal constant integer := 0;  
rowid_convert_external constant integer := 1;
```

例外

- ROWID\_INVALID

無効な ROWID 形式。
- ROWID\_BAD\_BLOCK

ブロックがファイルの最後を超えています。

次に例を示します。

```
ROWID_INVALID exception;  
pragma exception_init(ROWID_INVALID, -1410);  
  
ROWID_BAD_BLOCK exception;  
pragma exception_init(ROWID_BAD_BLOCK, -28516);
```

サブプログラムの要約

表 42-1 DBMS\_ROWID パッケージのサブプログラム

サブプログラム	説明
<a href="#">ROWID_CREATE</a> ファンクション (42-5 ページ)	ROWID をテスト用に限って作成します。
<a href="#">ROWID_INFO</a> プロシージャ (42-6 ページ)	ROWID のタイプとコンポーネントを戻します。
<a href="#">ROWID_TYPE</a> ファンクション (42-7 ページ)	ROWID のタイプを戻します。0 は制限付き、1 は拡張です。
<a href="#">ROWID_OBJECT</a> ファンクション (42-8 ページ)	拡張 ROWID のオブジェクト番号を戻します。
<a href="#">ROWID_RELATIVE_FNO</a> ファンクシ ョン (42-9 ページ)	ROWID のファイル番号を戻します。
<a href="#">ROWID_BLOCK_NUMBER</a> ファンクシ ョン (42-10 ページ)	ROWID のブロック番号を戻します。
<a href="#">ROWID_ROW_NUMBER</a> ファンクション (42-10 ページ)	行番号を戻します。
<a href="#">ROWID_TO_ABSOLUTE_FNO</a> ファンク ション (42-11 ページ)	特定の表にある行について、ROWID に関連する絶対ファ イル番号を戻します。
<a href="#">ROWID_TO_EXTENDED</a> ファンクション (42-12 ページ)	ROWID を制限形式から拡張形式に変換します。
<a href="#">ROWID_TO_RESTRICTED</a> ファンクシ ョン (42-14 ページ)	拡張 ROWID を制限形式に変換します。

表 42-1 DBMS\_ROWID パッケージのサブプログラム

サブプログラム	説明
<a href="#">ROWID_VERIFY</a> ファンクション (42-14 ページ)	ROWID_TO_EXTENDED ファンクションが ROWID を適切に拡張できるかどうかをチェックします。

## ROWID\_CREATE ファンクション

このファンクションによって、構成要素をパラメータとして ROWID を作成します。

Oracle Server では、データベース内のデータを指す有効な ROWID を作成することしかできないため、このファンクションは、ROWID 操作のテストに役立ちます。

### 構文

```
DBMS_ROWID.ROWID_CREATE (  
    rowid_type      IN NUMBER,  
    object_number   IN NUMBER,  
    relative_fno    IN NUMBER,  
    block_number    IN NUMBER,  
    row_number      IN NUMBER)  
RETURN ROWID;
```

### プラグマ

```
pragma RESTRICT_REFERENCES(rowid_create,WNDS,RNDS,WNPS,RNPS);
```

### パラメータ

表 42-2 ROWID\_CREATE ファンクションのパラメータ

パラメータ	説明
rowid_type	タイプ (制限または拡張)。  制限 ROWID については、rowid_type パラメータを 0 に設定します。拡張 ROWID を作成するには、1 を設定します。  rowid_type を 0 に指定すると、指定された object_number パラメータは無視され、ROWID_CREATE ファンクションは制限 ROWID を戻します。
object_number	データ・オブジェクト番号 (制限の場合は rowid_object_undefined)。
relative_fno	相対ファイル番号。
block_number	このファイル内のブロック番号。

表 42-2 ROWID\_CREATE ファンクションのパラメータ

パラメータ	説明
file_number	このブロック内のファイル番号。

例

ダミーの拡張 ROWID を作成します。

```
my_rowid := DBMS_ROWID.ROWID_CREATE(1, 9999, 12, 1000, 13);
```

rowid\_object ファンクションが戻す値を検索します。

```
obj_number := DBMS_ROWID.ROWID_OBJECT(my_rowid);
```

変数 obj\_number には、9999 が入っています。

ROWID\_INFO プロシージャ

このプロシージャは、ROWID のタイプ（制限または拡張）を含めた情報と、ROWID のコンポーネントを戻します。これはプロシージャで、SQL 文では使用できません。

構文

```
DBMS_ROWID.ROWID_INFO (  
    rowid_in          IN   ROWID,  
    rowid_type        OUT  NUMBER,  
    object_number     OUT  NUMBER,  
    relative_fno      OUT  NUMBER,  
    block_number      OUT  NUMBER,  
    row_number        OUT  NUMBER);
```

プラグマ

```
pragma RESTRICT_REFERENCES(rowid_info,WNDS,RNDS,WNPS,RNPS);
```

パラメータ

表 42-3 ROWID\_INFO プロシージャのパラメータ

パラメータ	説明
rowid_in	解釈する ROWID。ROWID が制限（0）か、または拡張（1）ROWID かを判別します。
rowid_type	タイプ（制限または拡張）を戻します。

表 42-3 ROWID\_INFO プロシージャのパラメータ

パラメータ	説明
object_number	データ・オブジェクト番号（制限の場合は rowid_object_undefined）を戻します。
relative_fno	相対ファイル番号を戻します。
block_number	このファイル内のブロック番号を戻します。
file_number	このブロック内のファイル番号を戻します。

関連項目：「[ROWID\\_TYPE ファンクション](#)」（42-7 ページ）

## 例

この例では、ROWID\_CREATE で作成した ROWID の値を読み込んで戻します。

```
DBMS_ROWID.ROWID_INFO(my_rowid, rid_type, obj_num,
    file_num, block_num, row_num);

DBMS_OUTPUT.PUT_LINE('The type is ' || rid_type);
DBMS_OUTPUT.PUT_LINE('Data object number is ' || obj_num);
-- and so on...
```

## ROWID\_TYPE ファンクション

このファンクションは、ROWID が制限 ROWID の場合は 0、拡張の場合は 1 を戻します。

## 構文

```
DBMS_ROWID.ROWID_TYPE (
    rowid_id IN ROWID)
RETURN NUMBER;
```

## プラグマ

```
pragma RESTRICT_REFERENCES(rowid_type,WNDS,RNDS,WNPS,RNPS);
```

## パラメータ

表 42-4 ROWID\_TYPE ファンクションのパラメータ

パラメータ	説明
row_id	解釈する ROWID。

例

```
IF DBMS_ROWID.ROWID_TYPE(my_rowid) = 1 THEN
  my_obj_num := DBMS_ROWID.ROWID_OBJECT(my_rowid);
```

ROWID\_OBJECT ファンクション

このファンクションは、拡張 ROWID のデータ・オブジェクト番号を戻します。入力された ROWID が制限 ROWID の場合は、0 を戻します。

構文

```
DBMS_ROWID.ROWID_OBJECT (
  rowid_id IN ROWID)
RETURN NUMBER;
```

プラグマ

```
pragma RESTRICT_REFERENCES(rowid_object,WNDS,RNDS,WNPS,RNPS);
```

パラメータ

表 42-5 ROWID\_OBJECT ファンクションのパラメータ

パラメータ	説明
row_id	解釈する ROWID。

---

---

**注意：** 制限 ROWID については、ROWID\_OBJECT\_UNDEFINED の定数が戻されます。

---

---

例

```
SELECT dbms_rowid.rowid_object(ROWID)
FROM emp
WHERE empno = 7499;
```

## ROWID\_RELATIVE\_FNO ファンクション

このファンクションは、IN パラメータで指定された ROWID の相対ファイル番号を戻します。(ファイル番号は表領域に関連しています。)

### 構文

```
DBMS_ROWID.ROWID_RELATIVE_FNO (  
    rowid_id IN ROWID)  
RETURN NUMBER;
```

### プラグマ

```
pragma RESTRICT_REFERENCES(rowid_relative_fno,WNDS,RNDS,WNPS,RNPS);
```

### パラメータ

表 42-6 ROWID\_RELATIVE\_FNO ファンクションのパラメータ

パラメータ	説明
row_id	解釈する ROWID。

### 例

次の例の PL/SQL コードは、相対ファイル番号を戻します。

```
DECLARE  
    file_number    INTEGER;  
    rowid_val      ROWID;  
BEGIN  
    SELECT ROWID INTO rowid_val  
    FROM dept  
    WHERE loc = 'Boston';  
    file_number :=  
        dbms_rowid.rowid_relative_fno(rowid_val);  
    ...
```

## ROWID\_BLOCK\_NUMBER ファンクション

このファンクションは、入力された ROWID のデータベース・ブロック番号を戻します。

### 構文

```
DBMS_ROWID.ROWID_BLOCK_NUMBER (  
    row_id IN ROWID)  
RETURN NUMBER;
```

### プラグマ

```
pragma RESTRICT_REFERENCES(rowid_block_number,WNDS,RNDS,WNPS,RNPS);
```

### パラメータ

表 42-7 ROWID\_BLOCK\_NUMBER ファンクションのパラメータ

パラメータ	説明
row_id	解釈する ROWID。

### 例

次の SQL 文の例では、ROWID からブロック番号を選択し、別の表に挿入します。

```
INSERT INTO T2 (SELECT dbms_rowid.rowid_block_number(ROWID)  
    FROM some_table  
    WHERE key_value = 42);
```

## ROWID\_ROW\_NUMBER ファンクション

このファンクションは、ROWID IN パラメータから行番号を抽出します。

### 構文

```
DBMS_ROWID.ROWID_ROW_NUMBER (  
    row_id IN ROWID)  
RETURN NUMBER;
```

### プラグマ

```
pragma RESTRICT_REFERENCES(rowid_row_number,WNDS,RNDS,WNPS,RNPS);
```



## パラメータ

表 42-8 ROWID\_ROW\_NUMBER ファンクションのパラメータ

パラメータ	説明
row_id	解釈する ROWID。

## 例

行番号を選択します。

```
SELECT dbms_rowid.rowid_row_number(ROWID)
FROM emp
WHERE ename = 'ALLEN';
```

## ROWID\_TO\_ABSOLUTE\_FNO ファンクション

このファンクションは、指定のスキーマと表にある行について、ファイル番号が絶対番号である ROWID から、絶対ファイル番号を抽出します。スキーマ名とスキーマ・オブジェクト名（表名など）がこのファンクションの IN パラメータとして提供されます。

## 構文

```
DBMS_ROWID.ROWID_TO_ABSOLUTE_FNO (
    row_id      IN ROWID,
    schema_name IN VARCHAR2,
    object_name IN VARCHAR2)
RETURN NUMBER;
```

## プラグマ

```
pragma RESTRICT_REFERENCES(rowid_to_absolute_fno,WNDS,WNPS,RNPS);
```

## パラメータ

表 42-9 ROWID\_TO\_ABSOLUTE\_FNO ファンクションのパラメータ

パラメータ	説明
row_id	解釈する ROWID。
schema_name	表を含んだスキーマの名前。
object_name	表名。

例

```
DECLARE
    abs_fno          INTEGER;
    rowid_val        CHAR(18);
    object_name      VARCHAR2(20) := 'EMP';
BEGIN
    SELECT ROWID INTO rowid_val
    FROM emp
    WHERE empno = 9999;
    abs_fno := dbms_rowid.rowid_to_absolute_fno(
        rowid_val, 'SCOTT', object_name);
```

**注意：** パーティション・オブジェクトの名前には、パーティション名やサブパーティション名ではなく、表名を指定する必要があります。

ROWID\_TO\_EXTENDED ファンクション

このファンクションは、ユーザーが指定するスキーマや表にある行をアドレス指定している制限形式の ROWID を、拡張 ROWID 形式に変換します。このファンクションは、今後、このパッケージから削除されて別の場所に移動する可能性があります。

構文

```
DBMS_ROWID.ROWID_TO_EXTENDED (
    old_rowid          IN ROWID,
    schema_name        IN VARCHAR2,
    object_name         IN VARCHAR2,
    conversion_type    IN INTEGER)
RETURN ROWID;
```

プラグマ

```
pragma RESTRICT_REFERENCES(rowid_to_extended,WNDS,WNPS,RNPS);
```

パラメータ

表 42-10 ROWID\_TO\_EXTENDED ファンクションのパラメータ

パラメータ	説明
old_rowid	変換する ROWID。
schema_name	表が含まれるスキーマの名前（オプション）。
object_name	表名（オプション）。

表 42-10 ROWID\_TO\_EXTENDED ファンクションのパラメータ

パラメータ	説明
conversion_type	rowid_convert_internal/external_convert_external (old_rowid が ROWID タイプの列に格納されていたか、または 文字列に格納されていたかによります)。

## 戻り値

ROWID\_TO\_EXTENDED は、拡張文字列形式で ROWID を戻します。入力された ROWID が NULL の場合、ファンクションは NULL を戻します。ゼロ値の ROWID ( 00000000.0000.0000 ) が提供されると、ゼロ値の制限 ROWID が戻されます。

## 例

SCOTT スキーマに RIDS と呼ばれる表があり、その表には、ROWID (制限) を保持する列 ROWID\_COL と、SCOTT スキーマのその他の表を指す列 TABLE\_COL が含まれていると仮定します。次の文を使用して、ROWID を拡張形式に変換できます。

```
UPDATE SCOTT.RIDS
SET rowid_col =
dbms_rowid.rowid_to_extended (
rowid_col, 'SCOTT', TABLE_COL, 0);
```

## 使用上の注意

スキーマ名とオブジェクト名が IN パラメータとして提供されると、このファンクションは、指定の表における SELECT 認可レベルを検証し、表のデータ・オブジェクト番号を使用して、提供された制限 ROWID を拡張 ROWID に変換します。ROWID\_TO\_EXTENDED は値を戻しますが、このファンクションがコールされたとき、または拡張 ROWID が実際に使用されたときに、変換された ROWID が表内の有効な行を実際に参照していることは保証しません。

スキーマ名とオブジェクト名が提供されない場合 ( NULL として渡された場合 ) このファンクションは、提供された制限 ROWID が指定しているページをフェッチしようとします。この ROWID に格納されているファイル番号は、絶対ファイル番号として処理されます。このため、そのファイルが削除されていたり、その番号が移動前に再使用されている場合は問題が生じます。フェッチされたページが有効な表に属している場合、この表のデータ・オブジェクト番号は拡張 ROWID 値への変換時に使用されます。これは非常に効率が悪いので、ターゲット表が不明な場合に行う最終手段としてのみお勧めします。ユーザーは、変換された値の使用時まで、正しい表名を覚えておく必要があります。

拡張 ROWID 値が提供された場合、入力した拡張 ROWID のデータ・オブジェクト番号は、表名パラメータから計算されたデータ・オブジェクト番号と照合して検証されます。2 つの番号が一致しない場合は、INVALID\_ROWID 例外が発生します。一致する場合は、入力した ROWID が戻されます。

**関連項目：**「[ROWID\\_VERIFY ファンクション](#)」には、指定の ROWID が拡張形式に変換できるかどうかを判別する方法が記述されています。

## ROWID\_TO\_RESTRICTED ファンクション

このファンクションは、拡張 ROWID を制限 ROWID 形式に変換します。

### 構文

```
DBMS_ROWID.ROWID_TO_RESTRICTED (  
    old_rowid          IN ROWID,  
    conversion_type IN INTEGER)  
RETURN ROWID;
```

### プラグマ

```
pragma RESTRICT_REFERENCES(rowid_to_restricted,WNDS,RNDS,WNPS,RNPS);
```

### パラメータ

表 42-11 ROWID\_TO\_RESTRICTED ファンクションのパラメータ

パラメータ	説明
old_rowid	変換する ROWID。
conversion_type	内部または外部 - 戻される ROWID の形式。  rowid_convert_internal/external_convert_external (戻される ROWID が ROWID タイプの列に格納されるか、または文 字列に格納されるかによります)。

## ROWID\_VERIFY ファンクション

このファンクションは、ROWID を検証します。入力した制限 ROWID が、入力スキーマ名と表名を指定して、拡張形式に変換できる場合は、0 を返し、変換できない場合は 1 を返します。

**注意：** 次の例に示すように、このファンクションを SQL 文の WHERE 句で使用できます。

構文

```
DBMS_ROWID.ROWID_VERIFY (  
    rowid_in      IN ROWID,  
    schema_name   IN VARCHAR2,  
    object_name   IN VARCHAR2,  
    conversion_type IN INTEGER  
    RETURN NUMBER;
```

プラグマ

```
pragma RESTRICT_REFERENCES(rowid_verify,WNDS,WNPS,RNPS);
```

パラメータ

表 42-12 ROWID\_VERIFY ファンクションのパラメータ

パラメータ	説明
rowid_in	検証する ROWID。
schema_name	表が含まれるスキーマの名前。
object_name	表名。
conversion_type	rowid_convert_internal/external_convert_external ( old_rowid が ROWID タイプの列に格納されていたか、または 文字列に格納されていたかによります )

例

ROWID\_TO\_EXTENDED ファンクションの例にあるスキーマを考慮しながら、次の文を使用すると、無効な ROWID を変換前に検索できます。これにより、無効を事前に修正できます。

```
SELECT ROWID, rowid_col  
FROM SCOTT.RIDS  
WHERE dbms_rowid.rowid_verify(rowid_col, NULL, NULL, 0) =1;
```

関連項目： [第 59 章「UTL\\_RAW」](#)、[第 60 章「UTL\\_REF」](#)



---

## DBMS\_SESSION

このパッケージは、PL/SQL から `SQL ALTER SESSION` 文と `SET ROLE` 文へのアクセスおよび他のセッション情報へのアクセスを提供します。このパッケージを使用すると、作業環境とセキュリティ・レベルを設定できます。

要件

このパッケージは、パッケージ所有者 SYS ではなく、コール・ユーザーの権限で実行されま  
す。

サブプログラムの要約

表 43-1 DBMS\_SESSION のサブプログラム

サブプログラム	説明
<a href="#">SET_ROLE プロシージャ</a> (43-3 ページ)	ロールを設定します。
<a href="#">SET_SQL_TRACE プロシージャ</a> (43-3 ページ)	トレースをオンまたはオフにします。
<a href="#">SET_NLS プロシージャ</a> (43-4 ページ)	各国語サポート (NLS) を設定します。
<a href="#">CLOSE_DATABASE_LINK プロシージャ</a> (43-4 ページ)	データベース・リンクをクローズします。
<a href="#">RESET_PACKAGE プロシージャ</a> (43-5 ページ)	セッション内のすべてのパッケージのインスタンス化の 解除をします。
<a href="#">UNIQUE_SESSION_ID ファンクション</a> (43-6 ページ)	データベースに現在接続しているすべてのセッションに 対して一意の識別子を戻します。
<a href="#">IS_ROLE_ENABLED ファンクション</a> (43-7 ページ)	指定のロールがセッションで使用可能かどうかを判別し ます。
<a href="#">IS_SESSION_ALIVE ファンクション</a> (43-7 ページ)	指定されたセッションが有効かどうかを判別します。
<a href="#">SET_CLOSE_CACHED_OPEN_CURSORS プロシージャ</a> (43-8 ページ)	close_cached_open_cursors をオンまたはオフにし ます。
<a href="#">FREE_UNUSED_USER_MEMORY プロシージャ</a> (43-8 ページ)	大量のメモリーが必要な操作を実行した後で未使用のメ モリーを再要求できます。
<a href="#">SET_CONTEXT プロシージャ</a> (43-11 ページ)	コンテキスト属性の値を設定または再設定します。
<a href="#">LIST_CONTEXT プロシージャ</a> (43-11 ページ)	現行セッションについて、アクティブな名前領域とコン テキストのリストを戻します。
<a href="#">SWITCH_CURRENT_CONSUMER_GROUP プロシージャ</a> (43-12 ページ)	ユーザーの現行セッションについて、現行リソース・コ ンシューマ・グループの変更を容易にします。



# SET\_ROLE プロシージャ

このプロシージャは、ロールを使用可能または使用禁止にします。このプロシージャは、SET\_ROLE SQL 文と同じです。

## 構文

```
DBMS_SESSION.SET_ROLE (  
    role_cmd VARCHAR2);
```

## パラメータ

表 43-2 SET\_ROLE プロシージャのパラメータ

パラメータ	説明
role_cmd	このテキストは "set role" に追加され、SQL として実行されます。

# SET\_SQL\_TRACE プロシージャ

このプロシージャは、トレースをオンまたはオフにします。このプロシージャは、次の SQL 文と同じです。

```
ALTER SESSION SET SQL_TRACE ...
```

## 構文

```
DBMS_SESSION.SET_SQL_TRACE (  
    sql_trace boolean);
```

## パラメータ

表 43-3 SET\_SQL\_TRACE プロシージャのパラメータ

パラメータ	説明
sql_trace	TRUE はトレースをオンにし、FALSE はトレースをオフにします。

## SET\_NLS プロシージャ

このプロシージャは、各国語サポート（NLS）を設定します。このプロシージャは、次の SQL 文と同じです。

```
ALTER SESSION SET <nls_parameter> = <value>
```

### 構文

```
DBMS_SESSION.SET_NLS (  
    param VARCHAR2,  
    value VARCHAR2);
```

### パラメータ

表 43-4 SET\_NLS プロシージャのパラメータ

パラメータ	説明
param	NLS パラメータ。このパラメータ名は、'NLS' で開始する必要があります。
value	パラメータ値。  パラメータがテキスト・リテラルの場合は、埋込みの一重引用符が必要です。たとえば、次のように指定します。"set_nls('nls_date_format','DD-MON-YY')"

## CLOSE\_DATABASE\_LINK プロシージャ

このプロシージャは、オープンしているデータベース・リンクをクローズします。このプロシージャは、次の SQL 文と同じです。

```
ALTER SESSION CLOSE DATABASE LINK <name>
```

### 構文

```
DBMS_SESSION.CLOSE_DATABASE_LINK (  
    dblink VARCHAR2);
```

### パラメータ

表 43-5 CLOSE\_DATABASE\_LINK プロシージャのパラメータ

パラメータ	説明
dblink	クローズするデータベース・リンク名。

## RESET\_PACKAGE プロシージャ

このプロシージャは、このセッションのすべてのパッケージのインスタンス化の解除をします。このプロシージャは、すべてのパッケージ状態を解放します。

実行状態をキャッシュするために使用するメモリーは、セッションで実行された PL/SQL ファンクション、プロシージャおよびパッケージに関連付けられています。

パッケージに関して、このメモリーのコレクションはパッケージ変数の現行値を保持し、各 PL/SQL プログラムがオープンしたカーソルのキャッシュを制御します。RESET\_PACKAGE へのコールは、以前実行した各 PL/SQL プログラムに関連付けられていたメモリーをセッションから解放します。したがって、グローバルなパッケージの現行値は消去され、キャッシュされたカーソルがクローズします。

RESET\_PACKAGE は、セッションで失敗したプログラムを確実に再起動するためにも使用できます。パッケージ変数を含んだプログラムが失敗すると、どの変数を初期化し直す必要があるかを判別することは困難です。RESET\_PACKAGE は、すべてのパッケージ変数が初期値に再設定されることを保証します。

### 構文

```
DBMS_SESSION.RESET_PACKAGE;
```

### パラメータ

なし。

### 使用上の注意

すべての実行 PL/SQL が消費するメモリーは大量になるため、RESET\_PACKAGE を使用して、データベース・アプリケーション内のある時点でセッション・メモリー・フットプリントを削減できます。ただし、パッケージ変数値の再設定がアプリケーションに影響を与えないことを確認してください。また、キャッシュしたメモリーとカーソルのないプログラムを後で実行すると、解放されたメモリーとカーソルを再作成する必要があるため、実行速度が遅くなることに留意してください。

RESET\_PACKAGE は、メモリー、カーソルおよびパッケージ変数をコール直後に解放しません。

---

**注意：** RESET\_PACKAGE は、起動した PL/SQL コールの実行が完了した後でのみ、メモリー、カーソルおよびパッケージ変数を解放します。

---

たとえば、PL/SQL プロシージャ P1 が PL/SQL プロシージャ P2 をコールし、P2 が RESET\_PACKAGE をコールしたとします。プロシージャ P1 の実行が完了するまで (PL/SQL コールが終了するまで)、RESET\_PACKAGE の処理は行われません。

例

SQL\*Plus スクリプトは、グローバル変数を使用する場合もしない場合もある多数の PL/SQL プログラム・ユニットを伴った大きいプログラムを実行します。ただし、実行後はグローバル変数は必要ありません。

```
EXECUTE large_plsql_program1;
```

キャッシュされた PL/SQL セッション・メモリーを解放します。

```
EXECUTE DBMS_SESSION.RESET_PACKAGE;
```

別の大きいプログラムを実行します。

```
EXECUTE large_plsql_program2;
```

UNIQUE\_SESSION\_ID ファンクション

このファンクションは、データベースに現在接続しているすべてのセッションに対して一意の識別子を戻します。同じセッション中にこのファンクションを複数回コールしても、常に同じ結果が戻されます。

構文

```
DBMS_SESSION.UNIQUE_SESSION_ID  
    RETURN VARCHAR2;
```

パラメータ

なし。

プログラマ

```
pragma restrict_references(unique_session_id,WNDS,RNDS,WNPS);
```

戻り値

表 43-6 UNIQUE\_SESSION\_ID ファンクションの戻り値

戻り値	説明
unique_session_id	最大 24 バイトまで戻します。

## IS\_ROLE\_ENABLED ファンクション

このファンクションは、指定のロールがこのセッションで使用可能かどうかを判別します。

### 構文

```
DBMS_SESSION.IS_ROLE_ENABLED (  
    rolename VARCHAR2)  
RETURN BOOLEAN;
```

### パラメータ

表 43-7 IS\_ROLE\_ENABLED ファンクションのパラメータ

パラメータ	説明
rolename	ロール名。

### 戻り値

表 43-8 IS\_ROLE\_ENABLED ファンクションの戻り値

戻り値	説明
is_role_enabled	ロールが使用可能かどうかによって TRUE または FALSE。

## IS\_SESSION\_ALIVE ファンクション

このファンクションは、指定されたセッションが有効かどうかを判別します。

### 構文

```
DBMS_SESSION.IS_SESSION_ALIVE (  
    uniqueid VARCHAR2)  
RETURN BOOLEAN;
```

### パラメータ

表 43-9 IS\_SESSION\_ALIVE ファンクションのパラメータ

パラメータ	説明
uniqueid	セッションの一意の ID。これは、UNIQUE_SESSION_ID で戻される ID と同じです。

戻り値

表 43-10 IS\_SESSION\_ALIVE ファンクションの戻り値

戻り値	説明
is_session_alive	セッションが有効かどうかによって TRUE または FALSE。

SET\_CLOSE\_CACHED\_OPEN\_CURSORS プロシージャ

このプロシージャは、close\_cached\_open\_cursors をオンまたはオフにします。このプロシージャは、次の SQL 文と同じです。

```
ALTER SESSION SET CLOSE_CACHED_OPEN_CURSORS ...
```

構文

```
DBMS_SESSION.SET_CLOSE_CACHED_OPEN_CURSORS (  
    close_cursors BOOLEAN);
```

パラメータ

表 43-11 SET\_CLOSE\_CACHED\_OPEN\_CURSORS プロシージャのパラメータ

パラメータ	説明
close_cursors	TRUE または FALSE。

FREE\_UNUSED\_USER\_MEMORY プロシージャ

このプロシージャは、大量のメモリー（100K を超えるメモリー）が必要な操作の実行後、未使用のメモリーを再要求します。

大量のメモリーを使用する操作の例を次に示します。

- sort\_area\_size 全部を使用し、sort\_area\_size が数百 K バイトになる大規模なソート処理。
- 大規模な PL/SQL パッケージ、プロシージャまたはファンクションのコンパイル処理。
- PL/SQL 索引表内にある数百 K バイトのデータを格納する処理。

V\$SESSTAT または V\$STATNAME の固定ビューにある統計情報 "session uga memory" と "session pga memory" を追跡調査して、ユーザー・メモリーをモニターできます。これらの統計情報のモニターでは、このプロシージャが解放したメモリー量も表示されます。

---

---

**注意：** このプロシージャは、メモリーが非常に不足している場合のみ使用してください。頻繁に使用せず、慎重に使用してください。

---

---

## 構文

```
DBMS_SESSION.FREE_UNUSED_USER_MEMORY;
```

## パラメータ

なし。

## 戻り値

このプロシージャの動作は、クライアントのかわりに稼働しているサーバーの構成によって決まります。

- **専用サーバー：**使用されていない PGA メモリーとセッション・メモリーをオペレーティング・システムに戻します。セッション・メモリーは、この構成内の PGA から割り当てられます。
- **MTS サーバー：**使用されていないセッション・メモリーを `shared_pool` に戻します。セッション・メモリーは、この構成内の `shared_pool` から割り当てられます。

## 使用上の注意

このプロシージャを使用してメモリーを解放するためには、そのメモリーが使用中でないことが必要です。

ある操作でメモリーを割り当てた後は、同じタイプの操作でのみ、割り当てられたメモリーを再使用できます。たとえば、ソート用にメモリーが割り当てられ、ソート完了後にそのメモリーが使用されなくなると、そのソート用に割り当てられたメモリーは別のソートでのみ再使用できます。ソートとコンパイルについては、操作の完了後にメモリーが使用されなくなると、ユーザーはこのプロシージャをコールして、この未使用メモリーを解放できます。

索引表にはメモリーが暗黙的に割り当てられ、その索引表の要素に割り当てられた値が格納されます。したがって、索引表内の要素が多いほど、RDBMS は多くのメモリーを索引表に割り当てます。索引表に要素がある間は、索引表に関連するメモリーは使用中になります。

索引表の有効範囲によって、メモリーの使用期間が決まります。グローバルに宣言された索引表は、パッケージまたはパッケージ本体で宣言された索引表です。これらの索引表には、セッション・メモリーからメモリーが割り当てられます。グローバルに宣言された索引表について、メモリーは、ユーザーのログイン中（ユーザーのセッションの間）は使用中のままとなり、Oracle から切断した後に解放されます。

ローカルで宣言された索引表は、ファンクション、プロシージャまたは無名ブロック内で宣言された索引表です。このような索引表には、PGA メモリーからメモリーが割り当てられます。ローカルで宣言された索引表について、索引表が宣言されているプロシージャ、ファ

ンクションまたは無名ブロックをユーザーが実行している間は、メモリーは使用中になります。プロシージャ、ファンクションまたは無名ブロックの実行が終了すると、そのメモリーは他のローカル宣言の索引表で使用可能になります（つまり、メモリーは使用中ではなくなります）。

初期化されていない空の索引表を既存の索引表に割り当てることは、索引表とその索引表に関連付けられたメモリーを明示的に再初期化するための1つの方法です。この操作を行うと、索引表に関連付けられているメモリーは使用中ではなくなり、このプロシージャをコールして解放できます。この方法は、グローバルに宣言した索引表がユーザー・セッションの期間中に大きくなる可能性があり、ユーザーが索引表の内容を必要としない場合は、特に役立ちます。

索引表の有効範囲に関連するメモリー・ルールは適用されたままです。しかし、この方法とプロシージャによって、ユーザーが介入して、索引表に関連付けられているメモリーを明示的に解放できます。

## 例

次の PL/SQL は、この方法と `FREE_UNUSED_USER_MEMORY` プロシージャの使用方法を示します。

```
CREATE PACKAGE foobar
    type number_idx_tbl is table of number indexed by binary_integer;

    store1_table number_idx_tbl;      -- PL/SQL indexed table
    store2_table number_idx_tbl;      -- PL/SQL indexed table
    store3_table number_idx_tbl;      -- PL/SQL indexed table
    ...
END;                                -- end of foobar

DECLARE
    ...
    empty_table number_idx_tbl;      -- uninitialized ("empty") version
BEGIN
    FOR i in 1..1000000 loop
        store1_table(i) := i;        -- load data
    END LOOP;
    ...
    store1_table := empty_table;      -- "truncate" the indexed table
    ...
    -
    dbms_session.free_unused_user_memory; -- give memory back to system

    store1_table(1) := 100;           -- index tables still declared;
    store2_table(2) := 200;           -- but truncated.
    ...
END;
```



## SET\_CONTEXT プロシージャ

このプロシージャは、コンテキスト属性の値を設定または再設定します。

### 構文

```
DBMS_SESSION.SET_CONTEXT (  
    namespace VARCHAR2,  
    attribute  VARCHAR2,  
    value      VARCHAR2);
```

### パラメータ

表 43-12 SET\_CONTEXT プロシージャのパラメータ

パラメータ	説明
namespace	アプリケーションのコンテキストで使用する名前領域の名前 (最大 30 バイト)。
attribute	設定する属性の名前 (最大 30 バイト)。
value	設定する値 (最大 256 バイト)。

### 使用上の注意

このファンクションのコール側は、CREATE CONTEXT 文を介してコンテキスト名前領域に関連付けられたプロシージャの呼出しスタックに存在している必要があります。呼出しスタックのチェックは、データベース管理システムの境界を越えません。

名前領域に設定できる属性の数に制限はありません。属性値は、ユーザーが再設定するまでユーザー・セッションの間そのまま残ります。

## LIST\_CONTEXT プロシージャ

このプロシージャは、現行セッションに関するアクティブな名前領域とコンテキストのリストを戻します。

### 構文

```
TYPE AppCtxRecTyp IS RECORD (  
    namespace VARCHAR2(30),  
    attribute  VARCHAR2(30),  
    value      VARCHAR2(256));  
  
TYPE AppCtxTabTyp IS TABLE OF AppCtxRecTyp INDEX BY BINARY_INTEGER;  
  
DBMS_SESSION.LIST_CONTEXT (  

```

```
list OUT AppCtxTabTyp,  
size OUT NUMBER);
```

パラメータ

表 43-13 LIST\_CONTEXT プロシージャのパラメータ

パラメータ	説明
list	現行セッション内のアプリケーション・コンテキスト設定リストを格納するバッファ。

戻り値

表 43-14 LIST\_CONTEXT プロシージャの戻り値

戻り値	説明
list	現行セッションにある設定（名前領域、属性、値）リスト。
size	戻されたバッファ内のエントリ数を戻します。

使用上の注意

リスト内のコンテキスト情報は、<namespace> <attribute> <value> の順に表示されます。list は表タイプの変数なので、そのサイズは戻されるリストのサイズに合わせて動的に調整されます。

SWITCH\_CURRENT\_CONSUMER\_GROUP プロシージャ

このプロシージャは、ユーザーの現行セッションでの現行リソース・コンシューマ・グループを変更します。

ある特定のグループに対してスイッチ権限がある場合は、コンシューマ・グループを切り替えることができます。コール側が別のプロシージャの場合、ユーザーは、そのプロシージャの所有者がスイッチ権限を持っているコンシューマ・グループに切り替えることができます。

構文

```
DBMS_SESSION.switch_current_consumer_group (  
  new_consumer_group      IN  VARCHAR2,  
  old_consumer_group      OUT VARCHAR2,  
  initial_group_on_error  IN   BOOLEAN);
```

## パラメータ

表 43-15 SWITCH\_CURRENT\_CONSUMER\_GROUP プロシージャのパラメータ

パラメータ	説明
new_consumer_group	切り替える先のコンシューマ・グループの名前。
old_consumer_group	切り替える元のコンシューマ・グループの名前。
initial_group_on_error	TRUE の場合は、エラー発生時にコール側の現行コンシューマ・グループが初期コンシューマ・グループとして設定されます。

## 戻り値

このプロシージャは、old\_consumer\_group パラメータにある、ユーザーの古いコンシューマ・グループを出力します。

**注意：** old\_consumer\_group で戻された値を使用して、古いコンシューマ・グループに後で切り替えることができます。

## 例外

表 43-16 SWITCH\_CURRENT\_CONSUMER\_GROUP プロシージャの例外

例外	説明
29368	コンシューマ・グループが存在しません。
1031	権限が不十分です。
29396	OTHER_GROUPS コンシューマ・グループに切り替えることができません。

## 使用上の注意

プロシージャの所有者には、ユーザーを古いコンシューマ・グループに再切替えるために、ユーザーの古いグループ (old\_consumer\_group) に対する権限が必要です。ただ 1 つの例外で、このプロシージャでは、ユーザーを初期のコンシューマ・グループにいつでも切り替えることができます (権限チェックはスキップ)。

initial\_group\_on\_error を TRUE に設定すると、SWITCH\_CURRENT\_CONSUMER\_GROUP プロシージャは、new\_consumer\_group が指定したグループに現行セッションを設定できない場合、そのセッションをデフォルト・グループに設定します。現行のコンシューマ・グループが初期のコンシューマ・グループに変更されていても、new\_consumer\_group へのセッションの移動に関連するエラーは発生します。

## 例

```
CREATE OR REPLACE PROCEDURE high_priority_task is
    old_group varchar2(30);
    prev_group varchar2(30);
    curr_user varchar2(30);
BEGIN
    -- switch invoker to privileged consumer group. If we fail to do so, an
    -- error will be thrown, but the consumer group will not change
    -- because 'initial_group_on_error' is set to FALSE

    dbms_session.switch_current_consumer_group('tkrogrp1', old_group, FALSE);
    -- set up exception handler (in the event of an error, we do not want to
    -- return to caller while leaving the session still in the privileged
    -- group)

    BEGIN
        -- perform some operations while under privileged group

    EXCEPTION
        WHEN OTHERS THEN
            -- It is possible that the procedure owner does not have privileges
            -- on old_group. 'initial_group_on_error' is set to TRUE to make sure
            -- that the user is moved out of the privileged group in such a
            -- situation

            dbms_session.switch_current_consumer_group(old_group,prev_group,TRUE);
            RAISE;
        END;

    -- we've succeeded. Now switch to old_group, or if cannot do so, switch
    -- to caller's initial consumer group

    dbms_session.switch_current_consumer_group(old_group,prev_group,TRUE);
END high_priority_task;
/
```

---

## DBMS\_SHARED\_POOL

DBMS\_SHARED\_POOL は、共有プールへのアクセスを提供します。この共有プールは、カーソルと PL/SQL オブジェクトが格納されている共有メモリー領域です。DBMS\_SHARED\_POOL によって、共有プール内のオブジェクト・サイズを表示したり、メモリーの断片化を減らすためにオブジェクトを保存または非保存としてマークすることができます。

## インストレーション時の注意

DBMS\_SHARED\_POOL を作成するには、DBMSPOOL.SQL スクリプトを実行します。  
PRVTPOOL.PLB スクリプトは、DBMSPOOL.SQL の実行後に自動的に実行されます。これらの  
スクリプトは、CATPROC.SQL では実行されません。

## 使用上の注意

ここで提供されるプロシージャは、大規模な PL/SQL オブジェクトのロード時に役立ちま  
す。大規模な PL/SQL オブジェクトのロード時には、大量の小規模オブジェクトを共有プー  
ルから期限切れとして削除して空き領域を用意する必要があるため（メモリー断片化のた  
め）、ユーザーの応答時間に影響を与えます。場合によっては、大規模なオブジェクトを  
ロードするためのメモリーが不足している場合があります。

DBMS\_SHARED\_POOL は、頻繁に実行するトリガーに対しても有効です。コンパイルしたト  
リガーを共有プールで頻繁に使用する表に保管できます。さらに、DBMS\_SHARED\_POOL は  
順序もサポートします。順序番号は、順序が期限切れで共有プールから削除されると失われ  
ます。DBMS\_SHARED\_POOL は、共有プールに順序を保存し、順序番号の損失を防止するの  
に役立ちます。

## サブプログラムの要約

表 44-1 DBMS\_SHARED\_POOL パッケージのサブプログラム

サブプログラム	説明
<a href="#">SIZES プロシージャ</a> ( 44-2 ページ )	共有プールにあるオブジェクトで、指定サイズより大きいオ ブジェクトを表示します。
<a href="#">KEEP プロシージャ</a> ( 44-3 ページ )	共有プールにオブジェクトを保存します。
<a href="#">UNKEEP プロシージャ</a> ( 44-5 ページ )	指定オブジェクトを解放します。
<a href="#">ABORTED_REQUEST_THRESHOLD プ ロシージャ</a> ( 44-5 ページ )	共有プールについて、異常終了を要求するしきい値を設定し ます。

## SIZES プロシージャ

このプロシージャは、shared\_pool にあるオブジェクトが指定したサイズより大きいオブ  
ジェクトを表示します。オブジェクト名を指定すると、後の KEEP または UNKEEP いずれか  
のコールへの引数として使用できます。

構文

```
DBMS_SHARED_POOL.SIZES (  
    minsize NUMBER);
```

パラメータ

表 44-2 SIZES プロシージャのパラメータ

パラメータ	説明
minsize	オブジェクトを表示するために、共有プール内で占有する必要があるサイズ（単位は K バイト）。

使用上の注意

このプロシージャの使用前に、SQL\*DBA または SQL\*PLUS 'SET SERVEROUTPUT ON SIZE XXXXX' コマンドを発行すると、結果が表示されます。

KEEP プロシージャ

このプロシージャは、共有プールにオブジェクトを保存します。オブジェクトが一度共有プールに保存されると、期間切れ削除の対象となりません。このプロシージャは、比較的頻繁に使用する大規模なオブジェクトに対して有効です。これは、大規模なオブジェクトが共有プールに移動されるときは、連続した十分な大きさの領域を作成するために、（移動されるオブジェクトのサイズよりはるかに大きい）他の大量のオブジェクトを期間切れで削除する必要があるためです。

構文

```
DBMS_SHARED_POOL.KEEP (  
    name VARCHAR2,  
    flag CHAR      DEFAULT 'P');
```

**注意：** このプロシージャは、自動メカニズムが将来インプリメントされて不要になった場合は、サポートされなくなる可能性があります。

## パラメータ

表 44-3 KEEP プロシージャのパラメータ

パラメータ	説明
name	保存するオブジェクトの名前。  この識別子は、アドレスと、v\$sqlarea ビューの hash_value 列を連結した値です。これは、SIZES プロシージャで表示され ます。  現在、TABLE と VIEW オブジェクトは保存できません。
flag	(オプション) このパラメータを指定しないと、パッケージは、最 初のパラメータをパッケージ、プロシージャまたはファンクシ ョンの名前とみなして名前を解決します。  入力内容がパッケージ、プロシージャまたはファンクションの名 前であることを完全に指定するには、'P' または 'p' を設定します。  入力内容がタイプ名であることを指定するには、'T' または 't' を設 定します。  入力内容がトリガー名であることを指定するには、'R' または 'r' を 設定します。  入力内容が順序名であることを指定するには、'Q' または 'q' を設 定します。  最初の引数がカーソル・アドレスとハッシュ値の場合、パラメー タには、'P' か 'p'、'Q' か 'q'、'R' か 'r'、'T' か 't' を除く任意の文字 を指定する必要があります。

## 例外

指定したオブジェクトが見つからない場合は、例外が発生します。

## 使用上の注意

オブジェクトは 2 種類あります。

- 名前で指定する PL/SQL オブジェクト、トリガー、順序およびタイプ。
- 2 つの番号（共有プール内の位置を示す）で指定する SQL カーソル・オブジェクト。

次に例を示します。

```
DBMS_SHARED_POOL.KEEP('scott.hispackage')
```

この例では、SCOTT が所有するパッケージ HISPACKAGE を保存します。PL/SQL オブジェ  
クトの名前は、オブジェクト命名の SQL ルールに従っています（つまり、区切られた名前  
やマルチバイトの名前などが可能です）。カーソルは、DBMS\_SHARED\_



POOL.KEEP('0034CDFF, 20348871') によって保存できます。最初の 8 文字は 16 進数の完全なアドレスである必要があります。

## UNKEEP プロシージャ

このプロシージャは、指定のオブジェクトを解放します。

### 構文

```
DBMS_SHARED_POOL.UNKEEP (  
    name VARCHAR2,  
    flag CHAR      DEFAULT 'P');
```

**注意：** このプロシージャは、自動メカニズムが将来インプリメントされて不要になった場合は、サポートされなくなる可能性があります。

### パラメータ

表 44-4 UNKEEP プロシージャのパラメータ

パラメータ	説明
name	保存しないオブジェクトの名前。「KEEP プロシージャ」の name パラメータの説明を参照してください。
flag	「KEEP プロシージャ」の flag パラメータの説明を参照してください。

### 例外

指定されたオブジェクトが見つからない場合は、例外が発生します。

## ABORTED\_REQUEST\_THRESHOLD プロシージャ

このプロシージャは、共有プールについて、異常終了を要求するしきい値を設定します。

### 構文

```
DBMS_SHARED_POOL.ABORTED_REQUEST_THRESHOLD (  
    threshold_size NUMBER);
```

パラメータ

表 44-5 ABORTED\_REQUEST\_THRESHOLD プロシージャのパラメータ

パラメータ	説明
threshold_size	共有プール内で確保解除されたメモリー（解放されたメモリーではない）を解放しないための要求サイズ（単位はバイト）。 threshold_size の範囲は 5000 ~ 2GB までです。

例外

しきい値が有効な範囲内にない場合は、例外が発生します。

使用上の注意

通常、要求が空きリストで満たされない場合、RDBMS は、LRU リストからオブジェクトを解放して要求を満たすことができるかを定期的にチェックし、メモリーを再要求しようとします。このステップの完了後、RDBMS は、'ALTER SYSTEM FLUSH SHARED\_POOL' とほぼ同等の内容を実行します。

これは、システム上のすべてのユーザーに影響を与えるため、このプロシージャは、その影響を thresh\_hold サイズを超える共有メモリーの断片検索に失敗した処理にローカライズします。このユーザーは、LRU リストを検索しなくても、'メモリー不足' エラーとなります。

---

## DBMS\_SNAPSHOT

DBMS\_SNAPSHOT によって、同じリフレッシュ・グループおよびバージ・ログの一部ではないスナップショットをリフレッシュできます。

---

**注意：** DBMS\_MVIEW は、DBMS\_SNAPSHOT のシノニムです。このシノニムは、データ・ウェアハウスで今後使用場合があります。

---

# サブプログラムの要約

表 45-1 DBMS\_SNAPSHOT パッケージのサブプログラム

サブプログラム	説明
<a href="#">BEGIN_TABLE_REORGANIZATION プロシージャ</a> (45-2 ページ)	リフレッシュに必要なスナップショット・データを保存するプロセスを実行します。
<a href="#">END_TABLE_REORGANIZATION プロシージャ</a> (45-3 ページ)	マスター表のスナップショット・データが有効であり、マスター表が適切な状態であることを確認します。
<a href="#">I_AM_A_REFRESH ファンクション</a> (45-3 ページ)	I_AM_REFRESH パッケージの状態の値を戻します。
<a href="#">PURGE_DIRECT_LOAD_LOG プロシージャ</a> (45-4 ページ)	スナップショットで行が不要になると、ダイレクト・ローダー・ログからその行を削除します (データ・ウェアハウスで使用)。
<a href="#">PURGE_LOG プロシージャ</a> (45-4 ページ)	スナップショット・ログから行をパージします。
<a href="#">PURGE_SNAPSHOT_FROM_LOG プロシージャ</a> (45-5 ページ)	スナップショット・ログから行をパージします。
<a href="#">REFRESH プロシージャ</a> (45-7 ページ)	スナップショットのリストをリフレッシュします。
<a href="#">REFRESH_ALL_MVIEWS プロシージャ</a> (45-9 ページ)	依存元表への最新の大量ロードのためにリフレッシュされなかったスナップショットをすべてリフレッシュします。
<a href="#">REFRESH_DEPENDENT プロシージャ</a> (45-10 ページ)	指定の元表または元表のリストに依存する表ベースのスナップショットをすべてリフレッシュします。
<a href="#">REGISTER_SNAPSHOT プロシージャ</a> (45-12 ページ)	個々のスナップショットの管理を可能にします。
<a href="#">UNREGISTER_SNAPSHOT プロシージャ</a> (45-13 ページ)	個々のスナップショットの管理を可能にします。マスター・サイトで起動して、スナップショットを登録解除します。

## BEGIN\_TABLE\_REORGANIZATION プロシージャ

このプロシージャは、リフレッシュに必要なスナップショット・データを保存するプロセスを実行します。このプロシージャは、マスター表の再編成前にコールする必要があります。

**関連項目：**『Oracle8i レプリケーション・ガイド』の「レプリケート環境の管理」を参照してください。

構文

```
DBMS_SNAPSHOT.BEGIN_TABLE_REORGANIZATION (  
    tabowner      IN   VARCHAR2,  
    tabname       IN   VARCHAR2);
```

パラメータ

表 45-2 BEGIN\_TABLE\_REORGANIZATION プロシージャのパラメータ

パラメータ	説明
tabowner	再編成する表の所有者。
tabname	再編成する表の名前。

END\_TABLE\_REORGANIZATION プロシージャ

このプロシージャは、マスター表の再編成後にコールする必要があります。マスター表のスナップショット・データが有効であり、マスター表が適切な状態であることを確認します。

**関連項目：**『Oracle8i レプリケーション・ガイド』の「レプリケート環境の管理」を参照してください。

構文

```
DBMS_SNAPSHOT.END_TABLE_REORGANIZATION (  
    tabowner      IN   VARCHAR2,  
    tabname       IN   VARCHAR2);
```

パラメータ

表 45-3 END\_TABLE\_REORGANIZATION プロシージャのパラメータ

パラメータ	説明
tabowner	再編成する表の所有者。
tabname	再編成する表の名前。

I\_AM\_A\_REFRESH ファンクション

このファンクションは、I\_AM\_REFRESH パッケージの状態の値を戻します。

構文

```
DBMS_SNAPSHOT.I_AM_A_REFRESH  
    RETURN BOOLEAN;
```

## パラメータ

なし。

## 戻り値

戻り値が `TRUE` の場合は、各レプリケーション・トリガーが最初にこの状態をチェックするため、このセッションではスナップショットのすべてのローカル・レプリケーション・トリガーが完全に使用禁止になります。戻り値が `FALSE` の場合は、そのトリガーを使用できません。

## PURGE\_DIRECT\_LOAD\_LOG プロシージャ

このプロシージャは、既知のスナップショット（マテリアライズド・ビュー）でエントリが不要になると、ダイレクト・ローダー・ログからそのエントリを削除します。このプロシージャは通常、Oracle のデータ・ウェアハウス・テクノロジーを使用する環境で使用されます。

**関連項目：** 詳細は、『Oracle8i チューニング』を参照してください。

## 構文

```
DBMS_SNAPSHOT.PURGE_DIRECT_LOAD_LOG;
```

## パラメータ

なし。

## PURGE\_LOG プロシージャ

このプロシージャは、スナップショット・ログから行を削除します。

## 構文

```
DBMS_SNAPSHOT.PURGE_LOG (  
    master      IN   VARCHAR2,  
    num         IN   BINARY_INTEGER := 1,  
    flag        IN   VARCHAR2      := 'NOP');
```

パラメータ

表 45-4 PURGE\_LOG プロシージャのパラメータ

パラメータ	説明
master	マスター表の名前。
num	<p>スナップショット・ログから行を削除したいスナップショットの中で、リフレッシュ日付が最も古いスナップショットの数。たとえば、次の文は、リフレッシュ日付が最も古い2つのスナップショットをリフレッシュするために必要な行を削除します。</p> <pre>dbms_snapshot.purge_log('master_table', 2);</pre> <p>スナップショット・ログにあるすべての行を削除するには、次の例のように、削除するスナップショットについて大きい数を指定します。</p> <pre>dbms_snapshot.purge_log('master_table', 9999);</pre> <p>この文は、MASTER_TABLE に基づくスナップショット数が 9999 未満の場合、MASTER_TABLE に対応しているスナップショット・ログを完全に削除します。行がスナップショット・ログからすでに削除されている単純なスナップショットは、次回リフレッシュ時に完全にリフレッシュする必要があります。</p>
flag	<p>スナップショット・ログから最低1つのスナップショットの行が削除されることを保証する場合は、DELETE を指定します。この引数は、引数 num の設定を上書きできます。たとえば、次の文は、スナップショット・ログに従属行が実際にあり、リフレッシュ日付が最も古いスナップショットから行を削除します。</p> <pre>dbms_snapshot.purge_log('master_table', 1, 'DELETE');</pre>

PURGE\_SNAPSHOT\_FROM\_LOG プロシージャ

このプロシージャは、スナップショット・リフレッシュに関連したデータ・ディクショナリ表にある行を削除するためにマスター・サイトでコールされます。この表は snapshot\_id または snapowner、snapname および snapsite の組合せで識別される指定スナップショットについて、マスター・サイトでメンテナンスされている表です。指定したスナップショットが、任意のマスター表からリフレッシュされた最も古いスナップショットの場合は、そのスナップショット・ログも削除されます。このプロシージャは、スナップショットの登録解除は行いません。

スナップショット・ログの1つを削除している間にエラーが発生した場合、それ以前に正常終了したスナップショット・ログの削除処理はロールバックされません。これは、スナップショット・ログのサイズを最小限にするためです。このプロシージャは、エラーが発生した場合でも、すべてのスナップショット・ログが削除されるまで再起動できます。

構文

```
DBMS_SNAPSHOT.PURGE_SNAPSHOT_FROM_LOG (  
    snapshot_id    IN    BINARY_INTEGER |  
    snapowner      IN    VARCHAR2,  
    snapname       IN    VARCHAR2,  
    snapsite       IN    VARCHAR2);
```

パラメータ

表 45-5 PURGE\_SNAPSHOT\_FROM\_LOG プロシージャのパラメータ

パラメータ	説明
snapshot_id	<p>このプロシージャをターゲット・スナップショットの ID に基づいて実行する場合は、snapshot_id パラメータでスナップショット ID を指定します。スナップショット ID のリストに関しては、スナップショット・サイトで DBA_SNAPSHOT_LOGS ビューを問い合わせます。</p> <p>登録済みスナップショットのリスト ( DBA_REGISTERED_SNAPSHOTS ) にターゲット・スナップショットがない場合は、スナップショット ID に基づいてこのプロシージャを実行すると便利です。</p> <p>スナップショット ID を指定する場合は、snapowner、snapname または snapsite パラメータに値を指定しないでください。</p>
snapowner	<p>snapshot_id を指定しない場合は、snapowner パラメータでターゲット・スナップショットの所有者を入力します。スナップショット・ログ・サイトで DBA_REGISTERED_SNAPSHOTS ビューを問い合わせ、スナップショットの所有者を表示します。</p>
snapname	<p>snapshot_id を指定しない場合は、snapname パラメータでターゲット・スナップショット名を入力します。スナップショット・ログ・サイトで DBA_REGISTERED_SNAPSHOTS ビューを問い合わせ、スナップショット名を表示します。</p>
snapsite	<p>snapshot_id を指定しない場合は、snapsite パラメータでターゲット・スナップショットのサイトを入力します。スナップショット・ログ・サイトで DBA_REGISTERED_SNAPSHOTS ビューを問い合わせ、スナップショットのサイトを表示します。</p>



# REFRESH プロシージャ

このプロシージャは、スナップショットのリストをリフレッシュします。

## 構文

```
DBMS_SNAPSHOT.REFRESH (
  { list          IN          VARCHAR2,
    | tab          IN OUT DBMS_UTILITY.UNCL_ARRAY, }
  method          IN          VARCHAR2      := NULL,
  rollback_seg     IN          VARCHAR2      := NULL,
  push_deferred_rpc IN          BOOLEAN      := TRUE,
  refresh_after_errors IN        BOOLEAN      := FALSE,
  purge_option      IN          BINARY_INTEGER := 1,
  parallelism       IN          BINARY_INTEGER := 0,
  heap_size         IN          BINARY_INTEGER := 0,
  atomic_refresh    IN          BOOLEAN      := TRUE);
```

## パラメータ

表 45-6 REFRESH プロシージャのパラメータ

パラメータ	説明
list tab	リフレッシュするスナップショットのカンマで区切られたリスト。 (シノニムはサポートされていません)。スナップショットは、異なるスキーマに配置したり、異なるマスター表を持つことができますが、リストされているすべてのスナップショットは、ローカル・データベースに存在する必要があります。他の方法として、DBMS_UTILITY.UNCL_ARRAY 型の PL/SQL 表を渡すことができます。このときの各要素がスナップショット名です。
method	リストされているスナップショットのリフレッシュ方法を示す文字列。 `F` または `f` は高速リフレッシュ、`?` は強制リフレッシュ、`C` または `c` は完全リフレッシュ、`A` または `a` は常によりフレッシュを示します。スナップショットに対応するリフレッシュ方法がない場合 (つまり、リフレッシュ方法より多くのスナップショットが指定された場合) そのスナップショットはデフォルトのリフレッシュ方法に従ってリフレッシュされます。たとえば、SQL*Plus 内で次の EXECUTE 文を実行します。  dbms_snapshot.refresh ('s_emp,s_dept,scott.s_salary','CF');  この文は、S_EMP スナップショットの完全リフレッシュ、S_DEPT スナップショットの高速リフレッシュ、および SCOTT.S_SALARY スナップショットのデフォルト・リフレッシュを実行します。
rollback_seg	スナップショットのリフレッシュ中に使用する、スナップショット・サイトのロールバック・セグメント名。

表 45-6 REFRESH プロシージャのパラメータ

パラメータ	説明
<code>push_deferred_rpc</code>	更新可能なスナップショットでのみ使用します。スナップショットのリフレッシュ前に、スナップショットから関連マスターに変更を送信する場合は、このパラメータを <code>TRUE</code> に設定します。そうでない場合は、変更が一時的に失われたように表示される場合があります。
<code>refresh_after_errors</code>	このパラメータが <code>TRUE</code> の場合は、スナップショットのマスター表の <code>DEFERROR</code> ビューに未解決の競合が記録されていても、更新可能なスナップショットのリフレッシュを続行します。このパラメータが <code>TRUE</code> で、 <code>atomic_refresh</code> が <code>FALSE</code> の場合、このプロシージャは、スナップショットのリフレッシュに失敗しても他のスナップショットのリフレッシュを続行します。
<code>purge_option</code>	パラレル伝播メカニズムを使用する場合（つまり、並列性に 1 以上を設定）、次のように指定します。0 = パージなし、1 = レイジー・パージ（デフォルト）、2 = アグレッシブ・パージ。ほとんどの場合、レイジー・パージが最適な設定です。複数のマスター・レプリケーション・グループが別々のターゲット・サイトに送信され、1 つ以上のレプリケーション・グループへの更新や送信がまれな場合は、アグレッシブ・パージに設定してキューを減らします。すべてのレプリケーション・グループへの更新と送信がまれな場合は、パージなしに設定し、キューを減らすために時々アグレッシブ・パージに設定して <code>PUSH</code> を実行してください。
<code>parallelism</code>	0 = シリアル伝播、 $n > 0 = n$ 個のパラレル・サーバー・プロセスを使用したパラレル伝播、1 = 1 つのパラレル・サーバー・プロセスのみを使用したパラレル伝播。
<code>heap_size</code>	パラレル伝播スケジューリングで同時に検査されるトランザクションの最大数。最適なパフォーマンスのためのデフォルト設定は Oracle が自動的に計算します。オラクル社カスタマ・サポートから指示がない限り、このパラメータは設定しないでください。
<code>atomic_refresh</code>	<p>このパラメータを <code>TRUE</code> に設定すると、スナップショットのリストは単一のトランザクションでリフレッシュされます。リフレッシュされたすべてのスナップショットは、同時に更新されます。スナップショットのいずれかでリフレッシュに失敗すると、すべてのスナップショットが更新されません。</p> <p>このパラメータを <code>FALSE</code> に設定すると、各スナップショットは個別のトランザクションでリフレッシュされます。このパラメータが <code>FALSE</code> の場合、ジョブ・キュー・プロセスの数は 1 以上に設定する必要があります。</p> <p><code>FALSE</code> の場合で、サマリー管理オプションが導入されていないと、エラーが発生します。</p>

## REFRESH\_ALL\_MVIEWS プロシージャ

このプロシージャは、次のプロパティを持つすべてのスナップショット（マテリアライズド・ビュー）をリフレッシュします。

- スナップショットが、依存している元表への最後の変更以降リフレッシュされていない場合。
- スナップショットとそれが依存しているすべての元表がローカルな場合。
- スナップショットが DBA\_MVIEW\_ANALYSIS ビューにある場合。

これは、データ・ウェアハウスで使用するためのプロシージャです。

### 構文

```
DBMS_SNAPSHOT.REFRESH_ALL_MVIEWS (  
    number_of_failures    OUT    BINARY_INTEGER,  
    method                IN     VARCHAR2          := NULL,  
    rollback_seg          IN     VARCHAR2          := NULL,  
    refresh_after_errors  IN     BOOLEAN           := FALSE,  
    atomic_refresh        IN     BOOLEAN           := TRUE);
```

### パラメータ

表 45-7 REFRESH\_ALL\_MVIEWS プロシージャのパラメータ

パラメータ	説明
number_of_failures	処理中に発生した失敗の件数を戻します。
method	各スナップショットに対して実行するリフレッシュのタイプを示す単一のリフレッシュ方法。`F' または `f' は高速リフレッシュ、`?' は強制リフレッシュ、`C' または `c' は完了リフレッシュ、`A' または `a' は常にリフレッシュを示します。方法が指定されない場合、スナップショットはデフォルトのリフレッシュ方法に従ってリフレッシュされます。
rollback_seg	スナップショットのリフレッシュ中に使用する、スナップショット・サイトのロールバック・セグメント名。
refresh_after_errors	このパラメータに TRUE を設定すると、スナップショットのマスター表の DEFERROR ビューに未解決の競合が記録されていても、更新可能なスナップショットのリフレッシュは続行します。このパラメータが TRUE で、atomic_refresh が FALSE の場合、このプロシージャは、スナップショットのリフレッシュに失敗しても他のスナップショットのリフレッシュを続行します。

表 45-7 REFRESH\_ALL\_MVIEWS プロシージャのパラメータ

パラメータ	説明
atomic_refresh	<p>このパラメータを TRUE に設定すると、リフレッシュされたスナップショットが単一のトランザクションでリフレッシュされます。リフレッシュされたすべてのスナップショットは、同時に更新されます。スナップショットのいずれかでリフレッシュに失敗すると、すべてのスナップショットが更新されません。</p> <p>このパラメータを FALSE に設定すると、リフレッシュされた各スナップショットが個別のトランザクションでリフレッシュされます。このパラメータが FALSE の場合、ジョブ・キュー・プロセスの数は 1 以上に設定する必要があります。</p> <p>FALSE の場合で、サマリー管理オプションが導入されていないと、エラーが発生します。</p>

REFRESH\_DEPENDENT プロシージャ

このプロシージャは、次のプロパティを持つすべてのスナップショット（マテリアライズド・ビュー）をリフレッシュします。

- スナップショットが指定の元表のリストにある元表に依存している場合。
- スナップショットが、依存している元表への最後の変更以降リフレッシュされていない場合。
- スナップショットとそれが依存しているすべての元表がローカルな場合。
- スナップショットが DBA\_MVIEW\_ANALYSIS ビューにある場合。

これは、データ・ウェアハウスで使用するためのプロシージャです。

構文

```
DBMS_SNAPSHOT.REFRESH_DEPENDENT (  
    number_of_failures    OUT    BINARY_INTEGER,  
    { list                IN     VARCHAR2,  
      | tab               IN OUT DBMS_UTILITY.UNCL_ARRAY, }  
    method               IN     VARCHAR2      := NULL,  
    rollback_seg         IN     VARCHAR2      := NULL,  
    refresh_after_errors IN     BOOLEAN       := FALSE,  
    atomic_refresh       IN     BOOLEAN       := TRUE);
```

## パラメータ

表 45-8 REFRESH\_DEPENDENT プロシージャのパラメータ

パラメータ	説明
number_of_failures	処理中に発生した失敗の件数を戻します。
list tab	スナップショットが依存できる元表のカンマで区切られたリスト。 (シノニムはサポートされていません)。これらの表とそれらに依存するスナップショットは、別々のスキーマに配置できます。ただし、すべての表とスナップショットは、ユーザーのローカル・データベースに存在している必要があります。他の方法として、DBMS_UTILITY.UNCL_ARRAY 型の PL/SQL 表を渡すことができます。このときの各要素は表の名前です。
method	リフレッシュ方法の文字列で、依存しているスナップショットのリフレッシュ方法を示します。特定の表に依存しているすべてのスナップショットは、その表に関連付けられたリフレッシュ方法に従ってリフレッシュされます。`F` または `f` は高速リフレッシュ、`?` は強制リフレッシュ、`C` または `c` は完了リフレッシュ、`A` または `a` は常にリフレッシュを示します。表に対応するリフレッシュ方法がない場合 (つまり、リフレッシュ方法より多くの表が指定された場合)、その表に依存するスナップショットはデフォルトのリフレッシュ方法に従ってリフレッシュされますたとえば、SQL*Plus 内の次の EXECUTE 文を実行します。  <pre>dbms_snapshot.refresh_dependent ('emp,dept,scott.salary','CF');</pre> この文は、EMP 表に依存するスナップショットの完全リフレッシュ、DEPT 表に依存するスナップショットの高速リフレッシュ、SCOTT.SALARY 表に依存するスナップショットのデフォルト・リフレッシュを実行します。
rollback_seg	スナップショットのリフレッシュ中に使用する、スナップショット・サイトのロールバック・セグメント名。
refresh_after_errors	このパラメータを TRUE に設定すると、スナップショットのマスター表の DEFERROR ビューに未解決の競合が記録されていても、更新可能なスナップショットのリフレッシュは続行します。このパラメータが TRUE で、atomic_refresh が FALSE の場合、このプロシージャは、スナップショットのリフレッシュに失敗しても他のスナップショットのリフレッシュを続行します。

表 45-8 REFRESH\_DEPENDENT プロシージャのパラメータ

パラメータ	説明
atomic_refresh	<p>このパラメータを <code>TRUE</code> に設定すると、リフレッシュされたスナップショットが単一のトランザクションでリフレッシュされます。リフレッシュされたすべてのスナップショットは、同時に更新されます。スナップショットのいずれかでリフレッシュに失敗すると、すべてのスナップショットが更新されません。</p> <p>このパラメータを <code>FALSE</code> に設定すると、リフレッシュされた各スナップショットが個別のトランザクションでリフレッシュされます。このパラメータが <code>FALSE</code> の場合、ジョブ・キュー・プロセスの数は 1 以上に設定する必要があります。</p> <p><code>FALSE</code> の場合で、サマリー管理オプションが導入されていないと、エラーが発生します。</p>

REGISTER\_SNAPSHOT プロシージャ

このプロシージャは、個々のスナップショットの管理を可能にします。

構文

```
DBMS_SNAPSHOT.REGISTER_SNAPSHOT (  
    snapowner    IN    VARCHAR2,  
    snapname     IN    VARCHAR2,  
    snapsite     IN    VARCHAR2,  
    snapshot_id  IN    DATE | BINARY_INTEGER,  
    flag         IN    BINARY_INTEGER,  
    qry_txt      IN    VARCHAR2,  
    rep_type     IN    BINARY_INTEGER := DBMS_SNAPSHOT.REG_UNKNOWN);
```

**注意：** このプロシージャはオーバーロードされています。snapshot\_id  
パラメータと flag パラメータは、両方同時には指定できません。

パラメータ

表 45-9 REGISTER\_SNAPSHOT プロシージャのパラメータ

パラメータ	説明
sowner	スナップショットの所有者。
snapname	スナップショット名。
snapsite	Oracle8 以降のマスター・サイトでスナップショットを登録ためのスナップショット・サイト名。このパラメータに二重引用符を含めることはできません。

表 45-9 REGISTER\_SNAPSHOT プロシージャのパラメータ

パラメータ	説明
snapshot_id	スナップショットの識別番号。Oracle8 のスナップショットは BINARY_INTEGER として指定します。Oracle8 以降のマスター・サイトで登録する Oracle7 のスナップショットは、DATE として指定します。
flag	PL/SQL パッケージ変数で、後続の作成または移動コマンドが問合せテキストに登録されているかどうかを示します。
query_txt	問合せの最初の 32,000 バイト。
rep_type	スナップショットのバージョン。代入可能な有効定数には、reg_unknown (デフォルト)、reg_v7_group、reg_v8_group および reg_repapi_group が含まれています。

使用上の注意

このプロシージャはマスター・サイトで実行され、リモート・プロシージャ・コールによって実行できます。REGISTER\_SNAPSHOT が同一の SNAPOWNER、SNAPNAME および SNAPSITE で繰り返しコールされる場合は、SNAPSHOT\_ID、FLAG および QUERY\_TXT の最新の値が格納されます。問合せが VARCHAR2 の最大サイズを超える場合は、最初の 32000 文字が QUERY\_TXT に格納され、残りは切り捨てられます。手動で起動するときは、プロシージャをコールするユーザーが、SNAPSHOT\_ID と FLAG の値をスナップショット・ビューで調べる必要があります。

スナップショットの問合せをマスター・サイトで登録しない場合は、オプションを FALSE に設定して SET\_REGISTER\_QUERY\_TEXT プロシージャをコールします。最新のオプション設定を調べるためには、DDL の発行前に、スナップショット・サイトで GET\_REG\_QUERY\_TEXT\_FLAG ファンクションをコールします。

UNREGISTER\_SNAPSHOT プロシージャ

このプロシージャは、個々のスナップショットの管理を可能にします。マスター・サイトで起動して、スナップショットを登録解除します。

構文

```
DBMS_SNAPSHOT.UNREGISTER_SNAPSHOT (  
    snapowner      IN   VARCHAR2,  
    snapname       IN   VARCHAR2,  
    snapsite       IN   VARCHAR2);
```

パラメータ

表 45-10 UNREGISTER\_SNAPSHOT プロシージャのパラメータ

パラメータ	説明
snapowner	スナップショットの所有者。
snapname	スナップショット名。
snapsite	スナップショット・サイト名。



---

## DBMS\_SPACE

DBMS\_SPACE パッケージによって、セグメントの成長と領域要件を分析できます。

## セキュリティ

このパッケージの実行には、SYS 権限が必要です。

## 要件

実行権限は、PUBLIC に付与されます。このパッケージのサブプログラムは、コール側のセキュリティ下で実行されます。ユーザーには、オブジェクトに関する ANALYZE 権限が必要です。

## サブプログラムの要約

表 46-1 DBMS\_SPACE パッケージのサブプログラム

サブプログラム	説明
<a href="#">UNUSED_SPACE プロシージャ</a> (46-2 ページ)	オブジェクト（表、索引またはクラスタ）にある未使用領域に関する情報を戻します。
<a href="#">FREE_BLOCKS プロシージャ</a> (46-3 ページ)	オブジェクト（表、索引またはクラスタ）にある空きブロックに関する情報を戻します。

## UNUSED\_SPACE プロシージャ

このプロシージャは、オブジェクト（表、索引またはクラスタ）にある未使用領域に関する情報を戻します。

### 構文

```
DBMS_SPACE.UNUSED_SPACE (  
    segment_owner          IN  VARCHAR2,  
    segment_name           IN  VARCHAR2,  
    segment_type           IN  VARCHAR2,  
    total_blocks           OUT NUMBER,  
    total_bytes            OUT NUMBER,  
    unused_blocks          OUT NUMBER,  
    unused_bytes           OUT NUMBER,  
    last_used_extent_file_id OUT NUMBER,  
    last_used_extent_block_id OUT NUMBER,  
    last_used_block        OUT NUMBER,  
    partition_name         IN  VARCHAR2 DEFAULT NULL);
```

パラメータ

表 46-2 UNUSED\_SPACE プロシージャのパラメータ

パラメータ	説明
segment_owner	分析するセグメントのスキーマ名。
segment_name	分析するセグメントのセグメント名。
segment_type	分析するセグメントのタイプは、次の中から選択します。 TABLE TABLE PARTITION TABLE SUBPARTITION INDEX INDEX PARTITION INDEX SUBPARTITION CLUSTER LOB
total_blocks	セグメント内のブロック合計数を戻します。
total_bytes	セグメント内のブロック合計数をバイト単位で戻します。
unused_blocks	未使用のブロック数を戻します。
unused_bytes	未使用のブロック数をバイト単位で戻します。
last_used_extent_ file_ id	データを含んだ最新エクステントのファイル ID を戻します。
last_used_extent_ block_ id	データを含んだ最新エクステントのブロック ID を戻します。
last_used_block	データを含んだエクステント内の最終ブロックを戻します。
partition_name	分析するセグメントのパーティション名。  これは、パーティション表についてのみ使用します。サブパーティションの名前は、パーティションの構成時に使用します。

FREE\_BLOCKS プロシージャ

このプロシージャは、オブジェクト（表、索引またはクラスタ）にある空きブロックに関する情報を戻します。

構文

```
DBMS_SPACE.FREE_BLOCKS (  
    segment_owner      IN  VARCHAR2,  
    segment_name       IN  VARCHAR2,  
    segment_type       IN  VARCHAR2,  
    freelist_group_id  IN  NUMBER,  
    free_blks          OUT NUMBER,  
    scan_limit         IN  NUMBER DEFAULT NULL,  
    partition_name     IN  VARCHAR2 DEFAULT NULL);
```

プラグマ

```
pragma restrict_references(free_blocks,WNDS);
```

パラメータ

表 46-3 FREE\_BLOCKS プロシージャのパラメータ

パラメータ	説明
segment_owner	分析するセグメントのスキーマ名。
segment_name	分析するセグメントのセグメント名。
segment_type	分析するセグメントのタイプは、次の中から選択します。 TABLE TABLE PARTITION TABLE SUBPARTITION INDEX INDEX PARTITION INDEX SUBPARTITION CLUSTER LOB
freelist_group_id	空きリスト・サイズが計算される空きリスト・グループ (インスタンス)。
free_blks	指定されたグループに関する空きブロック数を戻します。
scan_limit	読み込む空きリストのブロックの最大数 (オプション)。 空きリストに X 個のブロックがある場合は、X 個の走査制限を使用します。
partition_name	分析するセグメントのパーティション名。 これは、パーティション表についてのみ使用します。サブパーティションの名前は、パーティションの構成時に使用します。

## 例

### 例 1

次の例では、必要なバインド変数を宣言してから実行します。

```
DBMS_SPACE.UNUSED_SPACE('SCOTT', 'EMP', 'TABLE', :total_blocks,  
    :total_bytes, :unused_blocks, :unused_bytes, :lastextf,  
    :last_extb, :lastusedblock);
```

これにより、SCOTT スキーマの EMP 表に、バインド変数に関する未使用領域の情報が入ります。

### 例 2

次の例では、4 つの空きリスト・グループを持つ SCOTT スキーマにある CLUS クラスタが使用されます。そして、CLUS の空きリスト・グループ 3 にあるブロック数が戻されます。

```
DBMS_SPACE.FREE_BLOCKS('SCOTT', 'CLUS', 'CLUSTER', 3, :free_blocks);
```

---

**注意：** scan\_limit が正の数でない場合は、エラーが発生します。

---



---

## DBMS\_SPACE\_ADMIN

DBMS\_SPACE\_ADMIN パッケージは、ローカル管理の表領域に対する機能を提供します。

---

# セキュリティ

このパッケージは、SYS 権限で実行されるため、このパッケージを実行する権限を持つユーザーは、ビットマップも操作できます。

## 定数

SEGMENT_VERIFY_EXTENTS	セグメントが所有する領域の使用状況がビットマップに適切に反映されていることを検証します。
SEGMENT_VERIFY_EXTENTS_GLOBAL	セグメントが所有する領域の使用状況がビットマップに適切に反映されており、この領域が他のセグメントから要求されていないことを検証します。
SEGMENT_MARK_CORRUPT	一時セグメントを破損としてマークします。これにより、ディクショナリからの排除が容易になります（領域の再生なし）。
SEGMENT_MARK_VALID	破損した一時セグメントを有効としてマークします。これは、セグメントのエクステント・マップまたは他の場所での破損が解決され、セグメントが正常に削除できる場合に便利です。
SEGMENT_DUMP_EXTENT_MAP	指定のセグメントのエクステント・マップをダンプします。
TABLESPACE_VERIFY_BITMAP	表領域のビットマップを、その表領域内のセグメントのエクステント・マップを使用して検証し、すべてが一致していることを検証します。
TABLESPACE_EXTENT_MAKE_FREE	この範囲（エクステント）の領域をビットマップ上で空き領域にします。
TABLESPACE_EXTENT_MAKE_USED	この範囲（エクステント）の領域をビットマップ上で使用領域にします。



# サブプログラムの要約

表 47-1 DBMS\_SPACE\_ADMIN パッケージのサブプログラム

サブプログラム	説明
<a href="#">SEGMENT_VERIFY プロシージャ</a> (47-3 ページ)	セグメントのエクステント・マップの一貫性を検証します。
<a href="#">SEGMENT_CORRUPT プロシージャ</a> (47-4 ページ)	セグメントを破損または有効としてマークし、適切なエラーの回復を可能にします。
<a href="#">SEGMENT_DROP_CORRUPT プロシージャ</a> (47-5 ページ)	現在破損としてマークされているセグメントを削除します (領域の再生なし)。
<a href="#">SEGMENT_DUMP プロシージャ</a> (47-6 ページ)	指定セグメントのセグメント・ヘッダーとエクステント・マップをダンプします。
<a href="#">TABLESPACE_VERIFY プロシージャ</a> (47-7 ページ)	表領域内のセグメントについて、ビットマップとエクステント・マップが同期していることを検証します。
<a href="#">TABLESPACE_FIX_BITMAPS プロシージャ</a> (47-7 ページ)	適切な DBA 範囲 (エクステント) を、ビットマップ上で空きまたは使用領域としてマークします。
<a href="#">TABLESPACE_REBUILD_BITMAPS プロシージャ</a> (47-8 ページ)	適切なビットマップを再作成します。
<a href="#">TABLESPACE_REBUILD_QUOTAS プロシージャ</a> (47-9 ページ)	指定の表領域について割当て制限を再作成します。
<a href="#">TABLESPACE_MIGRATE_FROM_LOCAL プロシージャ</a> (47-10 ページ)	ローカル管理の表領域を、ディクショナリ管理の表領域に移行します。

## SEGMENT\_VERIFY プロシージャ

このプロシージャは、セグメントのエクステント・マップがビットマップと一致していることを検証します。

### 構文

```
DBMS_SPACE_ADMIN.SEGMENT_VERIFY (  
    tablespace_name      IN    VARCHAR2,  
    header_relative_file  IN    POSITIVE,  
    header_block          IN    POSITIVE,  
    verify_option         IN    POSITIVE DEFAULT SEGMENT_VERIFY_EXTENTS);
```

## パラメータ

表 47-2 SEGMENT\_VERIFY プロシージャのパラメータ

パラメータ	説明
tablespace_name	セグメントが常駐している表領域の名前。
header_relative_file	セグメント・ヘッダーの相対ファイル番号。
header_block	セグメント・ヘッダーのブロック番号。
verify_option	チェックの種類: SEGMENT_VERIFY_EXTENTS または SEGMENT_VERIFY_EXTENTS_GLOBAL。

## 使用上の注意

すべての DBA 範囲で誤った領域表現として検出された異常は、DBA 範囲、ビットマップ・ブロック、ビットマップ・ブロック範囲、異常情報としてトレース・ファイルに出力されます。レポートされる問題の種類は、空きとみなされない空き領域、空きとみなされた使用領域、および複数のセグメントで使用中とみなされた同一領域です。

## 例

次の例では、相対ファイル番号 4、ブロック番号 33 のセグメント・ヘッダーをもつセグメントで、エクステント・マップとビットマップが同期していることを検証します。

```
SQLPLUS > EXECUTE DBMS_SPACE_ADMIN.SEGMENT_VERIFY('USERS', 4, 33, 1);
```

**注意：** DBMS\_SPACE\_ADMIN パッケージのすべての例では、SCOTT.EMP を含んだ表領域 USERS が使用されます。

## SEGMENT\_CORRUPT プロシージャ

このプロシージャは、セグメントを破損または有効としてマークし、適切なエラーの回復を可能にします。

## 構文

```
DBMS_SPACE_ADMIN.SEGMENT_CORRUPT (
  tablespace_name      IN    VARCHAR2,
  header_relative_file  IN    POSITIVE,
  header_block         IN    POSITIVE,
  corrupt_option       IN    POSITIVE DEFAULT SEGMENT_MARK_CORRUPT);
```

パラメータ

表 47-3 SEGMENT\_CORRUPT プロシージャのパラメータ

パラメータ	説明
tablespace_name	セグメントが常駐している表領域の名前。
header_relative_file	セグメント・ヘッダーの相対ファイル番号。
header_block	セグメント・ヘッダーのブロック番号。
corrupt_option	SEGMENT_MARK_CORRUPT ( デフォルト ) または SEGMENT_MARK_VALID。

例

次の例では、セグメントを破損としてマークします。

```
SQLPLUS > EXECUTE DBMS_SPACE_ADMIN.SEGMENT_CORRUPT('USERS', 4, 33, 3);
```

次の例では、破損のセグメントを有効としてマークします。

```
SQLPLUS > EXECUTE DBMS_SPACE_ADMIN.SEGMENT_CORRUPT('USERS', 4, 33, 4);
```

SEGMENT\_DROP\_CORRUPT プロシージャ

このプロシージャは、現在破損としてマークされているセグメントを削除します（領域の再生なし）。これを実行するには、セグメントは一時的としてマークされている必要があります。破損のセグメントを一時的としてマークするには、セグメントで DROP コマンドを発行します。

セグメントのための領域は解放されないの、TABLESPACE\_FIX\_BITMAPS または TABLESPACE\_REBUILD\_BITMAPS プロシージャを使用して調整する必要があります。これについては、この章で後述します。

構文

```
DBMS_SPACE_ADMIN.SEGMENT_DROP_CORRUPT (
    tablespace_name      IN    VARCHAR2,
    header_relative_file  IN    POSITIVE,
    header_block          IN    POSITIVE);
```

パラメータ

表 47-4 SEGMENT\_DROP\_CORRUPT プロシージャのパラメータ

パラメータ	説明
tablespace_name	セグメントが常駐している表領域の名前。
header_relative_file	セグメント・ヘッダーの相対ファイル番号。
header_block	セグメント・ヘッダーのブロック番号。

例

```
SQLPLUS > EXECUTE DBMS_SPACE_ADMIN.SEGMENT_DROP_CORRUPT('USERS', 4, 33);
```

SEGMENT\_DUMP プロシージャ

このプロシージャは、指定のセグメントのセグメント・ヘッダーとエクステント・マップのブロックをダンプします。

構文

```
DBMS_SPACE_ADMIN.SEGMENT_DUMP (  
    tablespace_name      IN      VARCHAR2,  
    header_relative_file  IN      POSITIVE,  
    header_block          IN      POSITIVE,  
    dump_option           IN      POSITIVE DEFAULT SEGMENT_DUMP_EXTENT_MAP);
```

パラメータ

表 47-5 SEGMENT\_DUMP プロシージャのパラメータ

パラメータ	説明
tablespace_name	セグメントが常駐している表領域名。
header_relative_file	セグメント・ヘッダーの相対ファイル番号。
header_block	セグメント・ヘッダーのブロック番号。
dump_option	SEGMENT_DUMP_EXTENT_MAP

例

```
SQLPLUS > EXECUTE DBMS_SPACE_ADMIN.SEGMENT_DUMP('USERS', 4, 33);
```

# TABLESPACE\_VERIFY プロシージャ

このプロシージャは、表領域内のセグメントについて、ビットマップとエクステント・マップが同期していることを検証します。

## 構文

```
DBMS_SPACE_ADMIN.TABLESPACE_VERIFY (  
    tablespace_name      IN    VARCHAR2,  
    verify_option        IN    POSITIVE DEFAULT TABLESPACE_VERIFY_BITMAP);
```

## パラメータ

表 47-6 TABLESPACE\_VERIFY プロシージャのパラメータ

パラメータ	説明
tablespace_name	表領域名。
verify_option	TABLESPACE_VERIFY_BITMAP

## 例

```
SQLPLUS > EXECUTE DBMS_SPACE_ADMIN.TABLESPACE_VERIFY('USERS');
```

# TABLESPACE\_FIX\_BITMAPS プロシージャ

このプロシージャは、適切な DBA 範囲（エクステント）を、ビットマップ上で空きまたは使用領域としてマークします。

## 構文

```
DBMS_SPACE_ADMIN.TABLESPACE_FIX_BITMAPS (  
    tablespace_name      IN    VARCHAR2,  
    dbarange_relative_file IN    POSITIVE,  
    dbarange_begin_block IN    POSITIVE,  
    dbarange_end_block   IN    POSITIVE,  
    fix_option           IN    POSITIVE);
```

パラメータ

表 47-7 TABLESPACE\_FIX\_BITMAPS プロシージャのパラメータ

パラメータ	説明
tablespace_name	表領域の名前。
dbarange_relative_file	DBA 範囲 (エクステント) の相対ファイル番号。
dbarange_begin_block	エクステントの開始ブロック番号。
dbarange_end_block	エクステントの終了ブロック番号。
fix_option	TABLESPACE_EXTENT_MAKE_FREE または TABLESPACE_EXTENT_MAKE_USED。

例

次の例では、相対ファイル番号 4 の、ブロック番号 33 からブロック番号 83 までの 50 ブロックを、ビットマップ内 USED としてビットをマークします。

```
SQLPLUS > EXECUTE DBMS_SPACE_ADMIN.TABLESPACE_FIX_BITMAPS('USERS', 4, 33, 83, 7);
```

または、オプションに 8 を指定して、ビットマップ内のビットを FREE としてマークします。BEGIN と END ブロックは、エクステント境界内にあり、エクステントの倍数である必要があります。そうでない場合は、エラーが発生します。

TABLESPACE\_REBUILD\_BITMAPS プロシージャ

このプロシージャは、適切なビットマップを再作成します。ビットマップ・ブロックの DBA が指定されていない場合は、指定の表領域のすべてのビットマップが再作成されます。

構文

```
DBMS_SPACE_ADMIN.TABLESPACE_REBUILD_BITMAPS (  
    tablespace_name      IN      VARCHAR2,  
    bitmap_relative_file IN      POSITIVE  DEFAULT NULL,  
    bitmap_block         IN      POSITIVE  DEFAULT NULL);
```

パラメータ

表 47-8 TABLESPACE\_REBUILD\_BITMAPS プロシージャのパラメータ

パラメータ	説明
tablespace_name	表領域名。
bitmap_relative_file	再作成するビットマップ・ブロックの相対ファイル番号。
bitmap_block	再作成するビットマップ・ブロックのブロック番号。

例

次の例では、USERS 表領域にあるすべてのファイルについて、ビットマップを再作成します。

```
SQLPLUS > EXECUTE DBMS_SPACE_ADMIN.TABLESPACE_REBUILD_BITMAPS('USERS');
```

**注意：** 現在は、すべてのファイルの再作成のみサポートされています。

TABLESPACE\_REBUILD\_QUOTAS プロシージャ

このプロシージャは、指定の表領域に関する割当て制限を再作成します。

構文

```
DBMS_SPACE_ADMIN.TABLESPACE_REBUILD_QUOTAS (  
    tablespace_name          IN      VARCHAR2);
```

パラメータ

表 47-9 TABLESPACE\_REBUILD\_QUOTAS プロシージャのパラメータ

パラメータ	説明
tablespace_name	表領域の名前。

例

```
SQLPLUS > EXECUTE DBMS_SPACE_ADMIN.TABLESPACE_REBUILD_QUOTAS('USERS');
```

## TABLESPACE\_MIGRATE\_FROM\_LOCAL プロシージャ

このプロシージャは、ローカル管理の表領域をディクショナリ管理の表領域に移行します。

### 構文

```
DBMS_SPACE_ADMIN.TABLESPACE_MIGRATE_FROM_LOCAL (  
    tablespace_name          IN      VARCHAR2);
```

### パラメータ

表 47-10 TABLESPACE\_MIGRATE\_FROM\_LOCAL プロシージャのパラメータ

パラメータ	説明
tablespace_name	表領域の名前。

### 使用上の注意

表領域はオンラインで保持し、移行中に読取り書き込みを行う必要があります。  
テンポラリ表領域の移行と SYSTEM 表領域の移行はサポートされていません。

**注意：** SYSTEM 表領域の移行は、将来サポートされる予定です。

### 例

```
SQLPLUS > EXECUTE DBMS_SPACE_ADMIN.TABLESPACE_MIGRATE_FROM_LOCAL('USERS');
```

## 例

- TABLESPACE\_VERIFY で、一部のセグメントにビットマップ内空き領域としてマークされた割当てブロックがあるが、セグメント間オーバーラップについてはレポートされていないことが検出されました。この場合は、次のアクションをお勧めします。
  - SEGMENT\_EXTENT\_MAP\_DUMP プロシージャをコールして、このセグメントに割り当てられている DBA 範囲をダンプします。
  - 各範囲について、TABLESPACE\_MAKE\_USED オプションを指定した TABLESPACE\_FIX\_BITMAPS をコールして、その領域を使用領域としてマークします。
  - TABLESPACE\_REBUILD\_QUOTAS をコールして、割当て制限を修正します。
- ビットマップ内空き領域とマークされたセグメント・ブロックの一部があるため、セグメントを削除できません。このセグメントは破損として自動的にマークされています。この場合は、次の処理を行います。



- `SEGMENT_CHECK_ALL` オプションを指定した `SEGMENT_VERIFY` プロシージャをコールします。オーバーラップがレポートされていない場合は、次の処理を行います。
  - \* `SEGMENT_EXTENT_MAP_DUMP` プロシージャをコールして、セグメントに割り当てられている DBA 範囲をダンプします。
  - \* 各範囲について、`TABLESPACE_MAKE_FREE` オプションを指定した `TABLESPACE_FIX_BITMAPS` をコールして、その領域を空き領域としてマークします。
  - \* `SEGMENT_DROP_CORRUPT` をコールして、SEG\$ エントリを削除します。
  - \* `TABLESPACE_REBUILD_QUOTAS` をコールして、割当て制限を修正します。
- 3. `TABLESPACE_VERIFY` で、オーバーラップがレポートされました。この場合、以前の内部エラーに基づいて、実データの一部を破棄する必要がある場合があります。破棄するオブジェクトに表 T1 を選択してから、次の処理を行います。
  - T1 がオーバーラップしているすべてのオブジェクトのリストを作成します。
  - SQL を使用して、T1 を削除します。必要に応じて、`SEGMENT_DROP_CORRUPT` を後に続けます。
  - T1 がオーバーラップしていたすべてのオブジェクトについて、`SEGMENT_VERIFY` をコールします。必要に応じて、`TABLESPACE_FIX_BITMAPS` をコールして、適切なビットマップ・ブロックを使用領域としてマークします。
  - `TABLESPACE_VERIFY` を再実行します。
- 4. ビットマップ・ブロックの 1 セットがメディア破損です。次の処理を行います。
  - すべてのビットマップ・ブロックについて、または 1 つのブロックのみが破損している場合、単一のブロックについて、`TABLESPACE_REBUILD_BITMAPS` をコールします。
  - `TABLESPACE_REBUILD_QUOTAS` をコールして、割当て制限を再作成します。
  - `TABLESPACE_VERIFY` をコールして、ビットマップが一貫しているかどうかをチェックします。



Oracle によって、ユーザーは動的 SQL を使用するストアド・プロシージャと無名 PL/SQL ブロックを記述できます。動的 SQL 文は、ユーザーのソース・プログラムに埋め込まれていません。実行時にプログラムに入力されるか、またはプログラムによって作成されるように、文字列に格納されています。これによって、ユーザーは用途の広いプロシージャを作成できます。たとえば、この動的 SQL によって、実行時まで名前がわからない表で動作するプロシージャを作成できます。

また、データ操作言語 (DML) 文やデータ定義言語 (DDL) 文はいずれも、DBMS\_SQL パッケージを使用して解析できます。したがって、PL/SQL を使用して DDL 文を直接解析できます。たとえば、DBMS\_SQL パッケージが提供する PARSE プロシージャを使用することによって、ストアド・プロシージャ内から DROP TABLE 文の入力を選択できるようになりました。

---

**注意：** Oracle8i は、DBMS\_SQL パッケージのかわりにシステム固有の動的 SQL を導入しています。システム固有の動的 SQL を使用して、動的 SQL 文を PL/SQL ブロックに直接設定できます。

ほとんどの場合、DBMS\_SQL のかわりにシステム固有の動的 SQL を使用できます。システム固有の動的 SQL は、DBMS\_SQL と比べて使用方法が簡単でパフォーマンスが向上します。

---

**関連項目：** システム固有の動的 SQL の詳細は、『PL/SQL ユーザーズ・ガイドおよびリファレンス』を参照してください。

DBMS\_SQL とシステム固有の動的 SQL の比較は、『Oracle8i アプリケーション開発者ガイド 基礎編』を参照してください。

---

## DBMS\_SQL の使用方法

ストアド・プロシージャ内から動的 SQL を使用する機能は一般的に、Oracle コール・インタフェース (OCI) の手順に従っています。

**関連項目：**『Oracle8i コール・インタフェース・プログラマーズ・ガイド』

PL/SQL は、C などの他の一般的なプログラム言語とは、多少異なります。たとえば、ユーザーはアドレス (ポインタとも呼ばれる) を PL/SQL で参照できません。そのため、Oracle コール・インタフェースと DBMS\_SQL パッケージの間には、いくつか相違点があります。相違点は、次のとおりです。

- OCI はアドレスによるバインドを使用するのに対して、DBMS\_SQL パッケージは値によるバインドを使用します。
- DBMS\_SQL では、無名ブロックの OUT パラメータの値を検索するには、VARIABLE\_VALUE をコールする必要があります。また、行をフェッチして、行内の列の値を実際にプログラムに取り出した後で、COLUMN\_VALUE をコールする必要があります。
- 現在のリリースの DBMS\_SQL パッケージは、CANCEL カーソル・プロシージャを提供していません。
- NULL は PL/SQL 変数の値として完全にサポートされているため、標識変数は不要です。

DBMS\_SQL パッケージの使用例は、次のとおりです。このコードは、Oracle コール・インタフェースのユーザーにとってはかなり簡潔です。

### 例

この文のテキストはコンパイル時に判明しているため、この例では、動的 SQL を実際に使用する必要はありませんが、ここでは、このパッケージの概念をわかりやすく説明します。

DEMO プロシージャは、DEMO の実行時に指定した給与よりも高い給与のすべての従業員を EMP 表から削除します。

```
CREATE OR REPLACE PROCEDURE demo(salary IN NUMBER) AS
    cursor_name INTEGER;
    rows_processed INTEGER;
BEGIN
    cursor_name := dbms_sql.open_cursor;
    DBMS_SQL.PARSE(cursor_name, 'DELETE FROM emp WHERE sal > :x',
        dbms_sql.native);
    DBMS_SQL.BIND_VARIABLE(cursor_name, ':x', salary);
    rows_processed := dbms_sql.execute(cursor_name);
    DBMS_SQL.close_cursor(cursor_name);
EXCEPTION
    WHEN OTHERS THEN
        DBMS_SQL.close_cursor(cursor_name);
```

---

```
END;
```

## 定数

```
v6 constant INTEGER      := 0;
native constant INTEGER := 1;
v7 constant INTEGER      := 2;
```

## 型

```
TYPE varchar2s IS TABLE OF VARCHAR2(256) INDEX BY BINARY_INTEGER;
TYPE desc_rec  IS RECORD (
    col_type          BINARY_INTEGER := 0,
    col_max_len       BINARY_INTEGER := 0,
    col_name          VARCHAR2(32)   := '',
    col_name_len      BINARY_INTEGER := 0,
    col_schema_name   VARCHAR2(32)   := '',
    col_schema_name_len BINARY_INTEGER := 0,
    col_precision     BINARY_INTEGER := 0,
    col_scale         BINARY_INTEGER := 0,
    col_charsetid     BINARY_INTEGER := 0,
    col_charsetform   BINARY_INTEGER := 0,
    col_null_ok       BOOLEAN        := TRUE);
TYPE desc_tab IS TABLE OF desc_rec INDEX BY BINARY_INTEGER;
```

### 複数の SQL 型

```
type Number_Table IS TABLE OF NUMBER          INDEX BY BINARY_INTEGER;
type Varchar2_Table IS TABLE OF VARCHAR2(2000) INDEX BY BINARY_INTEGER;
type Date_Table    IS TABLE OF DATE           INDEX BY BINARY_INTEGER;
type Blob_Table    IS TABLE OF BLOB           INDEX BY BINARY_INTEGER;
type Clob_Table    IS TABLE OF CLOB           INDEX BY BINARY_INTEGER;
type Bfile_Table   IS TABLE OF BFILE          INDEX BY BINARY_INTEGER;
```

## 例外

```
inconsistent_type exception;
pragma exception_init(inconsistent_type, -6562);
```

この例外は、指定した OUT パラメータ（要求した値を設定するパラメータ）の型がその値の型と異なる場合に、プロシージャ COLUMN\_VALUE または VARIABLE\_VALUE で発生します。

---

## 実行フロー

### OPEN\_CURSOR

SQL 文を処理するためには、オープン・カーソルが必要です。OPEN\_CURSOR ファンクションをコールすると、ユーザーは Oracle が保持している有効なカーソルを表すデータ構造のカーソル ID 番号を受け取ります。これらのカーソルは、プリコンパイラ、OCI、または PL/SQL レベルで定義されたカーソルとは異なり、DBMS\_SQL パッケージでのみ使用されません。

### PARSE

SQL 文はすべて、PARSE プロシージャをコールして解析しなければなりません。文を解析することによって、その文の構文がチェックされ、プログラム内のカーソルに関連付けられます。

**関連項目：** SQL 文の解析方法は、『Oracle8i 概要』を参照してください。

DML 文または DDL 文はすべて解析できます。DDL 文は解析時に実行され、暗黙のコミットを実行します。

---

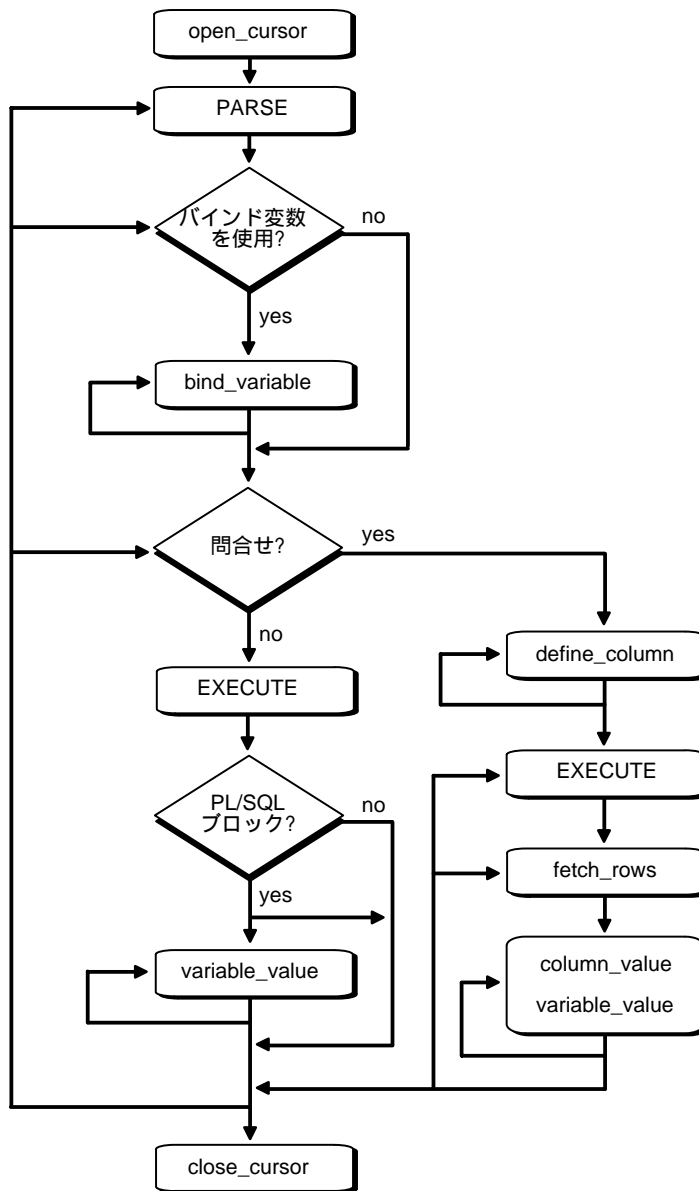
---

**注意：** パッケージやプロシージャを削除するために DDL 文を解析するときに、パッケージ内のプロシージャが使用中の場合はデッドロックが起こる可能性があります。プロシージャへのコール後は、実行がユーザー側に戻されるまで、そのプロシージャは使用中であるとみなされます。このようなデッドロックは、5 分後にタイムアウトします。

---

---

図 48-1 DBMS\_SQL 実行フロー



---

## **BIND\_VARIABLE または BIND\_ARRAY**

多くの DML 文では、プログラム内のデータを Oracle に入力することが必要です。実行時に提供する入力データを含んでいる SQL 文を定義する場合は、SQL 文内のプレースホルダを使用して、データの提供場所にマークを付ける必要があります。

SQL 文内の各プレースホルダに対してバインド・プロシージャ (BIND\_VARIABLE プロシージャまたは BIND\_ARRAY プロシージャ) の 1 つをコールして、プログラム内の変数の値 (または配列の値) をプレースホルダに提供する必要があります。SQL 文が引き続き実行されると、Oracle は、ユーザーのプログラムが入力変数と出力変数、またはバインド変数に設定したデータを使用します。

DBMS\_SQL は、その都度異なるバインド変数を使用して DML 文を繰り返し実行できます。BIND\_ARRAY プロシージャを使用すると、スカラーの集合をバインドでき、それぞれの値は 1 回の EXECUTE につき 1 度だけ入力変数として使用されます。これは、OCI がサポートする配列インタフェースに類似しています。

## **DEFINE\_COLUMN、DEFINE\_COLUMN\_LONG または DEFINE\_ARRAY**

SELECT 文内で選択されている行の列は、選択リスト内での相対位置 (左から右) によって識別されます。問合せの場合は、定義プロシージャの 1 つ (DEFINE\_COLUMN、DEFINE\_COLUMN\_LONG または DEFINE\_ARRAY) をコールして SELECT 値を受け入れる変数を指定する必要があります。これは、INTO 句が静的問合せに対して行う方法とほとんど同じです。

DEFINE\_COLUMN を使用して LONG 列以外の列を定義すると同様に、DEFINE\_COLUMN\_LONG プロシージャを使用して LONG 列を定義します。COLUMN\_VALUE\_LONG を使用して LONG 列からフェッチする前に、DEFINE\_COLUMN\_LONG をコールする必要があります。

DEFINE\_ARRAY プロシージャを使用して、行を単一の SELECT 文でフェッチする PL/SQL コレクションを定義します。DEFINE\_ARRAY は、1 回のフェッチで複数行をフェッチするインタフェースを提供します。COLUMN\_VALUE プロシージャで行をフェッチする前に、DEFINE\_ARRAY をコールする必要があります。

## **EXECUTE**

EXECUTE ファンクションをコールして、SQL 文を実行します。

## **FETCH\_ROWS または EXECUTE\_AND\_FETCH**

FETCH\_ROWS ファンクションは、問合せを満たす行を検索します。フェッチが行を検索できなくなるまで、連続する各フェッチは別の行を検索します。1 回だけの実行に対して EXECUTE をコールしている場合は、EXECUTE の次に FETCH\_ROWS をコールするより、EXECUTE\_AND\_FETCH をコールする方が効率的です。

## **VARIABLE\_VALUE、COLUMN\_VALUE または COLUMN\_VALUE\_LONG**

問合せの場合は、COLUMN\_VALUE をコールして、FETCH\_ROWS コールで検索する列の値を判別します。PL/SQL プロシージャへのコールまたは returning 句がある DML 文を含ん



---

だ無名ブロックの場合は、`VARIABLE_VALUE` をコールして、文の実行時に出力変数に割り当てられた値を検索します。

LONG データベース列（サイズは最大 2GB まで可能）の一部だけをフェッチするには、`COLUMN_VALUE_LONG` プロシージャを使用します。列値へのオフセット（単位はバイト）とフェッチするバイト数を指定できます。

## CLOSE\_CURSOR

セッションでカーソルが不要な場合は、`CLOSE_CURSOR` をコールしてカーソルをクローズします。Oracle Open Gateway を使用している場合は、これ以外のときにもカーソルのクローズが必要になる場合があります。追加情報は、Oracle Open Gateway の関連マニュアルを参照してください。

カーソルをクローズしないと、カーソルが不要になっても、そのカーソルが使用しているメモリーは割り当てられたままになります。

# セキュリティ

## 定義者権限モジュール

定義者権限モジュールは、モジュールの所有者の権限下で実行されます。定義者権限モジュールからコールされた `DBMS_SQL` サブプログラムは、モジュールで定義されたスキーマに関して実行されます。

---

**注意：** Oracle 8i より前のバージョンでは、すべての PL/SQL ストアド・プロシージャとパッケージが定義者権限モジュールでした。

---

## 実行者権限モジュール

実行者権限モジュールは、モジュールの実行者の権限下で実行されます。したがって、実行者権限モジュールからコールされた `DBMS_SQL` サブプログラムは、モジュールの実行者の権限下で実行されます。

モジュールに `current_user` に設定された `AUTHID` があると、未修飾の名前は、実行者のスキーマに関連付けて変換されます。

**例** `income` は `USER1` のスキーマにある実行者権限ストアド・プロシージャで、`USER2` には、そのストアド・プロシージャに対する `EXECUTE` 権限が付与されています。

```
CREATE PROCEDURE income(amount number)
  AUTHID current_user IS
  c number;
  n number;
BEGIN
  c:= dbms_sql.open_cursor;
```

```
dbms_sql.parse(c, 'insert into accts(''income'', :1)', dbms_sql.native);
dbms_sql.bind_variable(c, '1', amount);
n := dbms_sql.execute(c);
dbms_sql.close_cursor(c);
END;
```

USER1 が USER1.income をコールする場合は、USER1 の権限が使用され、未修飾名の名前変換が USER1 のスキーマに関して行われます。

USER2 が USER1.income をコールする場合は、USER2 の権限が使用され、未修飾名の名前変換（たとえば、accts）が USER2 のスキーマに関して行われます。

関連項目：『PL/SQL ユーザーズ・ガイドおよびリファレンス』

無名ブロック

無名 PL/SQL ブロックからコールされたすべての DBMS\_SQL サブプログラムは、現ユーザーの権限を使用して実行されます。

サブプログラムの要約

表 48-1 DBMS\_SQL パッケージのサブプログラム

サブプログラム	説明
<a href="#">OPEN_CURSOR</a> ファンクション (48-9 ページ)	新規カーソルのカーソル番号を戻します。
<a href="#">PARSE</a> プロシージャ (48-10 ページ)	指定した文を解析します。
<a href="#">BIND_VARIABLE</a> プロシージャ (48-12 ページ)	指定の値を指定の変数にバインドします。
<a href="#">BIND_ARRAY</a> プロシージャ (48-12 ページ)	指定の値を指定のコレクションにバインドします。
<a href="#">DEFINE_COLUMN</a> プロシージャ (48-17 ページ)	指定したカーソルから選択し、SELECT 文のみで使用する列を定義します。
<a href="#">DEFINE_ARRAY</a> プロシージャ (48-18 ページ)	指定したカーソルから選択し、SELECT 文のみで使用するコレクションを定義します。
<a href="#">DEFINE_COLUMN_LONG</a> プロシージャ (48-20 ページ)	指定したカーソルから選択し、SELECT 文のみで使用する LONG 列を定義します。
<a href="#">EXECUTE</a> ファンクション (48-20 ページ)	指定のカーソルを実行します。

表 48-1 DBMS\_SQL パッケージのサブプログラム

サブプログラム	説明
<a href="#">EXECUTE_AND_FETCH</a> ファンクション (48-21 ページ)	指定のカーソルを実行して、行をフェッチします。
<a href="#">FETCH_ROWS</a> ファンクション (48-21 ページ)	指定のカーソルから行をフェッチします。
<a href="#">COLUMN_VALUE</a> プロシージャ (48-22 ページ)	カーソルの指定位置にあるカーソル要素の値を戻します。
<a href="#">COLUMN_VALUE_LONG</a> プロシージャ (48-24 ページ)	DEFINE_COLUMN_LONG で定義した LONG 列の選択された部分を戻します。
<a href="#">VARIABLE_VALUE</a> プロシージャ (48-25 ページ)	指定のカーソルについて指定の変数の値を戻します。
<a href="#">IS_OPEN</a> ファンクション (48-27 ページ)	指定のカーソルがオープンの場合に TRUE を戻します。
<a href="#">DESCRIBE_COLUMNS</a> プロシージャ (48-28 ページ)	DBMS_SQL を介してオープンして解析されたカーソルの列を記述します。
<a href="#">CLOSE_CURSOR</a> プロシージャ (48-30 ページ)	指定したカーソルをクローズして、メモリーを解放します。
<a href="#">LAST_ERROR_POSITION</a> ファンクション (48-31 ページ)	エラーが発生した SQL 文テキスト内のバイト・オフセットを戻します。
<a href="#">LAST_ROW_COUNT</a> ファンクション (48-31 ページ)	フェッチされた累積行数を戻します。
<a href="#">LAST_ROW_ID</a> ファンクション (48-32 ページ)	最後に処理された行の ROWID を戻します。
<a href="#">LAST_SQL_FUNCTION_CODE</a> ファンクション (48-32 ページ)	文の SQL ファンクション・コードを戻します。

## OPEN\_CURSOR ファンクション

このプロシージャは、新規のカーソルをオープンします。このカーソルが不要になった場合は、CLOSE\_CURSOR をコールして、明示的にクローズする必要があります。

カーソルを使用すると、同じ SQL 文を繰り返し実行したり、新規の SQL 文を実行することができます。カーソルを再使用すると、対応するカーソル・データ領域の内容が新規 SQL 文の解析時に再設定されるため、再使用の前にクローズして再オープンする必要はありません。

## 構文

```
DBMS_SQL.OPEN_CURSOR  
    RETURN INTEGER;
```

## パラメータ

なし。

## プラグマ

```
pragma restrict_references(open_cursor,RNDS,WNDS);
```

## 戻り値

このファンクションは、新規カーソルのカーソル ID 番号を戻します。

## PARSE プロシージャ

このプロシージャは、指定したカーソル内の指定した文を解析します。すべての文が即時に解析されます。さらに、DDL 文は、解析時にただちに実行されます。

PARSE プロシージャには、引数として VARCHAR2 文を使用するバージョンと VARCHAR2S (VARCHAR2 の表) を使用するバージョンの 2 つがあります。

---

---

**注意：** DBMS\_SQL を使用して DDL 文を動的に実行すると、プログラムがハングする場合があります。たとえば、パッケージ内のプロシージャをコールすると、実行がユーザー側に戻るまでそのパッケージがロックされます。最初のロックを解放する前に、動的にパッケージを削除するなど、ロックの競合を引き起こす操作を行うとプログラムはハングします。

---

---

前述の構文を持つ SQL 文を解析するためのサイズは、32KB に制限されています。

## 構文

```
DBMS_SQL.PARSE (  
    c                      IN INTEGER,  
    statement              IN VARCHAR2,  
    language_flag          IN INTEGER);
```

PARSE プロシージャは、大規模な SQL 文について次の構文もサポートしています。

**注意：** このプロシージャは、PL/SQL 表の文の要素を連結し、その結果の文字列を解析します。このプロシージャを使用すると、文を分割することによって、単一の VARCHAR2 変数についての制限を超えた長い文を解析できます。

```
DBMS_SQL.PARSE (
  c              IN INTEGER,
  statement      IN VARCHAR2S,
  lb             IN INTEGER,
  ub             IN INTEGER,
  lfflg         IN BOOLEAN,
  language_flag  IN INTEGER);
```

パラメータ

表 48-2 PARSE プロシージャのパラメータ

パラメータ	説明
c	文を解析するカーソルの ID 番号。
statement	解析する SQL 文。  PL/SQL 文と異なり、SQL 文の終わりにはセミコロンを含めないでください。次に例を示します。  DBMS_SQL.PARSE(cursor1, 'BEGIN proc; END;', 2); DBMS_SQL.PARSE(cursor1, 'INSERT INTO tab values(1)', 2);
lb	文内の要素の下限。
ub	文内の要素の上限。
lfflg	TRUE の場合、連結している各要素の後に改行を挿入します。
language_flag	Oracle で SQL 文を処理する方法を決定します。次のオプションが認識されます。 <ul style="list-style-type: none"><li>■ V6 (または 0) は、バージョン 6 の動作を指定します。</li><li>■ NATIVE (または 1) は、プログラムの接続先のデータベースに関する通常の動作を指定します。</li><li>■ V7 (または 2) は、Oracle7 の動作を指定します。</li></ul>

---

---

**注意：** クライアント側コードは、リモート・パッケージの変数または定数を参照できないため、定数の値を明示的に使用する必要があります。

たとえば、次のコードは、クライアント側でコンパイルしません。

```
DBMS_SQL.PARSE(cur_hdl, stmt_str, dbms_sql.V7); -- uses constant
dbms_sql.V7
```

次のコードは、引数が明示的に指定されているので、クライアント側で有効です。

```
DBMS_SQL.PARSE(cur_hdl, stmt_str, 2); -- compiles on the client
```

---

---

**大規模な SQL 文字列を解析するための VARCHAR2S データ型** 32KB を超える SQL 文を解析するために、DBMS\_SQL パッケージは、PL/SQL 表を使用して文字列の表を PARSE プロシージャに渡します。これらの文字列は連結された後、Oracle Server に渡されます。

ローカル変数を VARCHAR2S 表項目型として宣言し、次に、PARSE プロシージャを使用すると、大規模な SQL 文を VARCHAR2S として解析できます。

VARCHAR2S データ型の定義は、次のとおりです。

```
TYPE varchar2s IS TABLE OF VARCHAR2(256) INDEX BY BINARY_INTEGER;
```

## 例外

コンパイル警告を持つ DBMS\_SQL を使用して、型、プロシージャ、ファンクションまたはパッケージを作成すると、ORA-24344 例外が発生しますが、プロシージャはそのまま作成されます。

## BIND\_VARIABLE プロシージャ

## BIND\_ARRAY プロシージャ

この 2 つのプロシージャは、文内の変数の名前に基づいて、カーソル内の指定の変数に指定の値または値のセットをバインドします。この変数が IN 変数、IN/OUT 変数、または IN コレクションである場合は、指定したバインド値が、変数タイプまたは配列タイプに対して有効である必要があります。OUT 変数のバインド値は無視されます。

SQL 文のバインド変数またはコレクションは、名前によって識別されます。バインド変数またはバインド配列に値をバインドする場合は、次の例に示すように、文中でバインド変数を識別する文字列の先頭にコロンを付ける必要があります。

```
SELECT emp_name FROM emp WHERE SAL > :X;
```

この例では、対応するバインド・コールは次のようになります。

```
BIND_VARIABLE(cursor_name, ':X', 3500);
```

or

```
BIND_VARIABLE (cursor_name, 'X', 3500);
```

## 構文

```
DBMS_SQL.BIND_VARIABLE (  
    c                IN INTEGER,  
    name             IN VARCHAR2,  
    value            IN <datatype>);
```

<datatype> は、次のいずれかの型である必要があります。

```
NUMBER  
DATE  
VARCHAR2 CHARACTER SET ANY_CS  
BLOB  
CLOB CHARACTER SET ANY_CS  
BFILE
```

BIND\_VARIABLE は、異なるデータ型を受け入れるために二重定義されていることに注意してください。

**関連項目：**『Oracle8i アプリケーション開発者ガイド ラージ・オブジェクト』

## プラグマ

```
pragma restrict_references(bind_variable,WNDS);
```

## 使用上の注意

BIND\_VARIABLE では、次の構文もサポートされています。大カッコ [] は、BIND\_VARIABLE ファンクションのオプション・パラメータを示します。

```
DBMS_SQL.BIND_VARIABLE (  
    c                IN INTEGER,  
    name             IN VARCHAR2,  
    value            IN VARCHAR2 CHARACTER SET ANY_CS [,out_value_size IN INTEGER]);
```

CHAR、RAW および ROWID データをバインドするために、次のパリエーションを構文で使用できます。

```
DBMS_SQL.BIND_VARIABLE_CHAR (  
    c                IN INTEGER,  
    name             IN VARCHAR2,
```

```
value          IN CHAR CHARACTER SET ANY_CS [,out_value_size IN INTEGER]);

DBMS_SQL.BIND_VARIABLE_RAW (
  c             IN INTEGER,
  name          IN VARCHAR2,
  value         IN RAW [,out_value_size IN INTEGER]);

DBMS_SQL.BIND_VARIABLE_ROWID (
  c             IN INTEGER,
  name          IN VARCHAR2,
  value         IN ROWID);
```

パラメータ

表 48-3 BIND\_VARIABLE プロシージャのパラメータ

パラメータ	説明
c	値をバインドするカーソルの ID 番号。
name	文内の変数の名前。
value	カーソル内の変数にバインドする値。  IN 変数と IN/OUT 変数の場合、この値は、このパラメータで渡される値の型と同じ型です。
out_value_size	VARCHAR2、RAW、CHAR OUT または IN/OUT 変数の最大予測 OUT 値サイズ（単位はバイト）。  サイズの指定がない場合は、現行値の長さが使用されます。このパラメータは、value パラメータが初期化されていない場合、指定する必要があります。

一括配列バインド

一括選択、挿入、更新および削除は、多くのコールを 1 つにまとめることによって、アプリケーションのパフォーマンスを改善できます。DBMS\_SQL パッケージによって、ユーザーは PL/SQL 表タイプを使用しながらデータの収集に対する処理を実行できます。

表項目は、バインドされていない同種のコレクションです。表項目は、持続記憶領域では他の関係表に似ており、組込みの配列を持ちません。ただし、表項目が、（問合せまたは持続データのナビゲーション・アクセスのいずれかによって）作業領域に移されたり、あるいは PL/SQL の変数またはパラメータの値として作成されると、要素の値を取得して設定するために配列形式の構文で利用できる添え書きが、その表項目の要素に与えられます。

これらの要素の添え書きは詳細である必要はなく、負数を含むあらゆる数値が使用できます。たとえば、表項目には、-10、2 および 7 の位置だけにある要素を含めることができます。



表項目が一時作業領域から持続記憶領域に移されると、添え書きは格納されません。つまり、表項目は、持続記憶領域内では順序が付いていません。

表は、バインド実行時に、PL/SQL バッファからローカルの DBMS\_SQL バッファにコピーされ（すべてのスカラー型について同様）、ローカルの DBMS\_SQL バッファから操作されます。したがって、バインド・コール後に表を変更した場合でも、その変更が実行方法に影響を与えることはありません。

## スカラー型と LOB コレクション型

ローカル変数を次のいずれかの表項目型として宣言できます。これらの表項目型は、DBMS\_SQL ではパブリック・タイプとして定義されています。

```
type Number_Table  IS TABLE OF NUMBER          INDEX BY BINARY_INTEGER;
type Varchar2_Table IS TABLE OF VARCHAR2(2000) INDEX BY BINARY_INTEGER;
type Date_Table    IS TABLE OF DATE            INDEX BY BINARY_INTEGER;
type Blob_Table     IS TABLE OF BLOB            INDEX BY BINARY_INTEGER;
type Clob_Table     IS TABLE OF CLOB            INDEX BY BINARY_INTEGER;
type Bfile_Table    IS TABLE OF BFILE           INDEX BY BINARY_INTEGER;
```

## 構文

```
DBMS_SQL.BIND_ARRAY (
    c                IN INTEGER,
    name             IN VARCHAR2,
    <table_variable> IN <datatype>
    [,index1         IN INTEGER,
    index2           IN INTEGER]) );
```

<table\_variable> とそれに対応する <datatype> は、次のいずれかの組合せになります。

```
<num_tab>      Number_Table
<vchr2_tab>    Varchar2_Table
<date_tab>     Date_Table
<blob_tab>     Blob_Table
<clob_tab>     Clob_Table
<bfile_tab>    Bfile_Table
```

BIND\_ARRAY プロシージャは、異なるデータ型を受け入れるために二重定義されていることに注意してください。

## パラメータ

表 48-4 BIND\_ARRAY プロシージャのパラメータ

パラメータ	説明
c	値をバインドするカーソルの ID 番号。
name	文内のコレクションの名前。
table_variable	<datatype> として宣言されたローカル変数。
index1	範囲の下限を示す表要素の索引。
index2	範囲の上限を示す表要素の索引。

### 使用上の注意

範囲をバインドするためには、範囲を指定する要素（タブ（index1）とタブ（index2））が表に含まれている必要がありますが、その範囲は詳細でなくても構いません。index1 には、index2 以下の値を指定してください。タブ（index1）とタブ（index2）の間にあるすべての要素がバインドして使用されます。

バインド・コールで索引を指定しない場合で、かつ文内の 2 つの異なるバインドが異なる数の要素を含んだ表を指定している場合、実際に使用される要素の数は、すべての表の最小値となります。これは索引を指定する場合にも当てはまります。つまり、すべての表に関する 2 つの索引の間では最小範囲が選択されます。

問合せ内のすべてのバインド変数が、配列バインドである必要はありません。一部は通常のバインドの場合があり、式の評価などでは、同じ値がコレクションの各要素に使用されません。

**関連項目：** コレクションのバインド方法の例は、「[例 3、4 および 5: 一括 DML](#)」（48-35 ページ）を参照してください。

## 問合せの処理

動的 SQL を使用して問合せを処理する場合は、次のステップを実行しなければなりません。

1. DEFINE\_COLUMN、DEFINE\_COLUMN\_LONG または DEFINE\_ARRAY をコールして、SELECT 文が戻す値を受け入れる変数を指定します。
2. EXECUTE をコールして、SELECT 文を実行します。
3. FETCH\_ROWS（または EXECUTE\_AND\_FETCH）をコールして、問合せに合致した行を検索します。
4. 問合せに関して FETCH\_ROWS コールが検索した列の値を判別するために、COLUMN\_VALUE または COLUMN\_VALUE\_LONG をコールします。PL/SQL プロシージャへのコー

ルを含んだ無名ブロックを使用した場合は、`VARIABLE_VALUE` をコールして、これらのプロシージャの出力変数に割り当てられた値を検索します。

## DEFINE\_COLUMN プロシージャ

このプロシージャは、指定のカーソルから選択する列を定義します。このプロシージャが使用できるのは、`SELECT` カーソルのみです。

定義されている列は、指定のカーソル内にある文の `SELECT` リスト内での相対位置によって識別されます。`COLUMN` 値の型によって、定義される列の型が決まります。

### 構文

```
DBMS_SQL.DEFINE_COLUMN (
    c                IN INTEGER,
    position         IN INTEGER,
    column           IN <datatype>);
```

<datatype> は、次のいずれかの型である必要があります。

```
NUMBER
DATE
BLOB
CLOB CHARACTER SET ANY_CS
BFILE
```

`DEFINE_COLUMN` は、異なるデータ型を受け入れるために二重定義されていることに注意してください。

**関連項目：**『Oracle8i アプリケーション開発者ガイド ラージ・オブジェクト』

### プラグマ

```
pragma restrict_references(define_column,RNDS,WNDS);
```

`DEFINE_COLUMN` プロシージャでは、次の構文もサポートされます。

```
DBMS_SQL.DEFINE_COLUMN (
    c                IN INTEGER,
    position         IN INTEGER,
    column           IN VARCHAR2 CHARACTER SET ANY_CS,
    column_size      IN INTEGER);
```

`CHAR`、`RAW` および `ROWID` データを持つ列の定義には、プロシージャ構文で次のパリエーションを使用できます。

```
DBMS_SQL.DEFINE_COLUMN_CHAR (
```

```
c                IN INTEGER,
position         IN INTEGER,
column           IN CHAR CHARACTER SET ANY_CS,
column_size      IN INTEGER);

DBMS_SQL.DEFINE_COLUMN_RAW (
  c                IN INTEGER,
  position         IN INTEGER,
  column           IN RAW,
  column_size      IN INTEGER);

DBMS_SQL.DEFINE_COLUMN_ROWID (
  c                IN INTEGER,
  position         IN INTEGER,
  column           IN ROWID);
```

パラメータ

表 48-5 DEFINE\_COLUMN プロシージャのパラメータ

パラメータ	説明
c	選択対象に定義されている行のカーソルの ID 番号。
position	定義している行内にある列の相対位置。 文の最初の列は位置 1 です。
column	定義している列の値。 この値の型によって、定義している列の型が決まります。
column_size	VARCHAR2 型、CHAR 型、および RAW 型の列に対する列値の最大予測サイズ（単位はバイト）。

DEFINE\_ARRAY プロシージャ

このプロシージャは、( FETCH\_ROWS コールで ) 行をフェッチする列に対してコレクションを定義します。このプロシージャによって、ユーザーは単一の SELECT 文から、行を一括してフェッチできます。1 回のフェッチ・コールで、PL/SQL の集計オブジェクトに多数の行をフェッチできます。

行をフェッチすると、それらの行は COLUMN\_VALUE コールを実行するまで DBMS\_SQL バッファにコピーされ、COLUMN\_VALUE コールの実行時点で、引数として渡された表にコピーされます。

## コレクション用のスカラー型と LOB 型

ローカル変数を、次のいずれかの表項目型として宣言し、DBMS\_SQL を使用して、任意の行数をその中にフェッチできます。(これらの表項目型は、BIND\_ARRAY プロシージャに指定できる型と同じです)。

```
type Number_Table    IS TABLE OF NUMBER          INDEX BY BINARY_INTEGER;
type Varchar2_Table  IS TABLE OF VARCHAR2(2000) INDEX BY BINARY_INTEGER;
type Date_Table      IS TABLE OF DATE            INDEX BY BINARY_INTEGER;
type Blob_Table      IS TABLE OF BLOB             INDEX BY BINARY_INTEGER;
type Clob_Table      IS TABLE OF CLOB             INDEX BY BINARY_INTEGER;
type Bfile_Table     IS TABLE OF BFILE           INDEX BY BINARY_INTEGER;
```

## 構文

```
DBMS_SQL.DEFINE_ARRAY (
    c           IN INTEGER,
    position    IN INTEGER,
    bf_tab      IN Bfile_Table,
    cnt         IN INTEGER,
    lower_bound IN INTEGER);
```

## プラグマ

```
pragma restrict_references(define_array,RNDS,WNDS);
```

後続の FETCH\_ROWS コールが count 行をフェッチします。COLUMN\_VALUE がコールされると、これらの行は位置 indx、indx+1、indx+2 のように配置されます。行が送られてくる間、ユーザーは FETCH\_ROWS コールまたは COLUMN\_VALUE コールを継続して発行します。行は、COLUMN\_VALUE コールに引数として指定した表内に蓄積されます。

## パラメータ

表 48-6 DEFINE\_ARRAY プロシージャのパラメータ

パラメータ	説明
c	配列をバインドするカーソルの ID 番号。
position	定義している配列内にある列の相対位置。 文の最初の列は位置 1 です。
column	このパラメータに渡される値の型は、定義される列の型です。
column_size	VARCHAR2 列の値の最大予測サイズ (単位はバイト)。

count にはゼロより大きい整数を指定してください。それ以外の値が指定されると、例外が発生します。indx は、正の数、負の数またはゼロでも構いません。DEFINE\_ARRAY コールが発行された問合せに、配列バインドを含めることはできません。

**関連項目：** コレクションの定義方法の例は、「[例 6 および 7: 配列の定義](#)」(48-37 ページ)を参照してください。

## DEFINE\_COLUMN\_LONG プロシージャ

このプロシージャは、SELECT カーソルに対して LONG 列を定義します。定義されている列は、指定のカーソルの文の SELECT リスト内での相対位置によって識別されます。COLUMN 値の型によって、定義されている列の型が決まります。

### 構文

```
DBMS_SQL.DEFINE_COLUMN_LONG (  
    c                IN INTEGER,  
    position         IN INTEGER);
```

### パラメータ

表 48-7 DEFINE\_COLUMN\_LONG プロシージャのパラメータ

パラメータ	説明
c	選択対象に定義されている行のカーソルの ID 番号。
position	定義している行内にある列の相対位置。 文の最初の列は位置 1 です。

## EXECUTE ファンクション

このファンクションは、指定のカーソルを実行します。このファンクションはカーソルの ID 番号を受け入れて、処理された行数を戻します。戻り値は、DDL 文を含めて、INSERT 文、UPDATE 文、および DELETE 文に対してのみ有効で、戻り値が未定義の場合は無視されます。

### 構文

```
DBMS_SQL.EXECUTE (  
    c    IN INTEGER)  
RETURN INTEGER;
```

## パラメータ

表 48-8 EXECUTE ファンクションのパラメータ

パラメータ	説明
c	実行するカーソルのカーソル ID 番号。

## EXECUTE\_AND\_FETCH ファンクション

このファンクションは、指定のカーソルを実行して行をフェッチします。このファンクションは、EXECUTE をコールしてから FETCH\_ROWS をコールするのと同じ機能を提供します。ただし、リモート・データベースで使用する場合は、EXECUTE\_AND\_FETCH をコールした方がネットワークのラウンドトリップ数を低減できます。

EXECUTE\_AND\_FETCH ファンクションは、実際にフェッチされた行数を戻します。

## 構文

```
DBMS_SQL.EXECUTE_AND_FETCH (  
    c                IN INTEGER,  
    exact            IN BOOLEAN DEFAULT FALSE)  
RETURN INTEGER;
```

## プラグマ

```
pragma restrict_references(execute_and_fetch,WNDS);
```

## パラメータ

表 48-9 EXECUTE\_AND\_FETCH ファンクションのパラメータ

パラメータ	説明
c	実行してフェッチするカーソルのカーソル ID 番号。
exact	問合せで実際に一致する行数が 1 以外の場合は、TRUE に設定すると、例外が発生します。  例外が発生しても、行はフェッチされ、使用可能です。

## FETCH\_ROWS ファンクション

このファンクションは、指定のカーソルから行をフェッチします。FETCH\_ROWS は、フェッチする行が残っている限り、繰り返しコールできます。これらの行はバッファに取り出し、FETCH\_ROWS への各コール後に、COLUMN\_VALUE をコールして各列ごとに読み込む必要があります。

FETCH\_ROWS ファンクションは、フェッチするカーソルの ID 番号を受け入れて、実際にフェッチされた行数を戻します。

構文

```
DBMS_SQL.FETCH_ROWS (  
    c                IN INTEGER)  
    RETURN INTEGER;
```

パラメータ

表 48-10 FETCH\_ROWS ファンクションのパラメータ

パラメータ	説明
c	ID 番号。

プラグマ

```
pragma restrict_references(fetch_rows,WNDS);
```

COLUMN\_VALUE プロシージャ

このプロシージャは、指定したカーソル内の指定の位置にあるカーソル要素の値を戻します。このプロシージャは、FETCH\_ROWS をコールしてフェッチしたデータへのアクセスに使用されます。

構文

```
DBMS_SQL.COLUMN_VALUE (  
    c                IN INTEGER,  
    position         IN INTEGER,  
    value            OUT <datatype>  
    [,column_error   OUT NUMBER]  
    [,actual_length  OUT INTEGER]);
```

<datatype> は、次のいずれかの型である必要があります。

- NUMBER
- DATE
- VARCHAR2 CHARACTER SET ANY\_CS
- BLOB
- CLOB CHARACTER SET ANY\_CS
- BFILE



---

**注意：** 大カッコ [] は、オプション・パラメータを示します。

---

**関連項目：** 『Oracle8i アプリケーション開発者ガイド ラージ・オブジェクト』

## プラグマ

```
pragma restrict_references(column_value,RNDS,WNDS);
```

COLUMN\_VALUE プロシージャでは、次の構文もサポートされます。

```
DBMS_SQL.COLUMN_VALUE(
    c                IN  INTEGER,
    position         IN  INTEGER,
    <table_variable> IN  <datatype>);
```

<table\_variable> とそれに対応する <datatype> は、次のいずれかの組合せが可能です。

```
<num_tab>      Number_Table
<vchr2_tab>    Varchar2_Table
<date_tab>     Date_Table
<blob_tab>     Blob_Table
<clob_tab>     Clob_Table
<bfile_tab>    Bfile_Table
```

CHAR、RAW および ROWID データを含んだ列では、次のバリエーションを構文で使用できます。

```
DBMS_SQL.COLUMN_VALUE_CHAR (
    c                IN  INTEGER,
    position         IN  INTEGER,
    value            OUT CHAR CHARACTER SET ANY_CS
    [,column_error   OUT NUMBER]
    [,actual_length  OUT INTEGER]);
```

```
DBMS_SQL.COLUMN_VALUE_RAW (
    c                IN  INTEGER,
    position         IN  INTEGER,
    value            OUT RAW
    [,column_error   OUT NUMBER]
    [,actual_length  OUT INTEGER]);
```

```
DBMS_SQL.COLUMN_VALUE_ROWID (
```

```
c                IN  INTEGER,
position         IN  INTEGER,
value            OUT ROWID
[,column_error   OUT NUMBER]
[,actual_length  OUT INTEGER]);
```

パラメータ

表 48-11 COLUMN\_VALUE プロシージャのパラメータ

パラメータ	説明
c	値をフェッチするカーソルの ID 番号。
position	カーソル内の列の相対位置。 文の最初の列は位置 1 です。
value	指定した列と行の値を戻します。  指定した行番号がフェッチされた行数の合計より大きい場合は、エラー・メッセージが発生します。  この出力パラメータの型が、DEFINE_COLUMN へのコールで定義されている値の実際の型と異なる場合、例外エラー ORA-06562、inconsistent_type が発生します。
table_variable	<datatype> として宣言されたローカル変数。
column_error	指定した列値のエラー・コードを戻します。
actual_length	指定した列内の値の（切捨て前の）実際の長さ。

**例外：**  
指定した OUT パラメータの value が、実際の値の型と異なる場合は、inconsistent\_type (ORA-06562) が発生します。この型は、DEFINE\_COLUMN プロシージャをコールして列を定義したときに指定した型です。

COLUMN\_VALUE\_LONG プロシージャ

このプロシージャは、LONG 列の値の一部を取得します。

構文

```
DBMS_SQL.COLUMN_VALUE_LONG (
  c                IN  INTEGER,
  position         IN  INTEGER,
  length          IN  INTEGER,
  offset          IN  INTEGER,
```

```
value          OUT VARCHAR2,  
value_length  OUT INTEGER);
```

プラグマ

```
pragma restrict_references(column_value_long,RNDS,WNDS);
```

パラメータ

表 48-12 COLUMN\_VALUE\_LONG プロシージャのパラメータ

パラメータ	説明
c	値を取得するカーソルのカーソル ID 番号。
position	値を取得する列の位置。
length	フェッチする LONG 値のバイト数。
offset	フェッチを開始するための LONG フィールドへのオフセット。
value	VARCHAR2 の列の値。
value_length	値に実際に戻されるバイト数。

VARIABLE\_VALUE プロシージャ

このプロシージャは、指定のカーソルについて指定の変数の値を戻します。このプロシージャは、`returning` 句を使用して PL/SQL ブロックまたは DML 文内のバインド変数の値を戻すために使用されます。

構文

```
DBMS_SQL.VARIABLE_VALUE (  
    c          IN  INTEGER,  
    name       IN  VARCHAR2,  
    value      OUT <datatype>);
```

<datatype> は、次のいずれかの型である必要があります。

```
NUMBER  
DATE  
VARCHAR2 CHARACTER SET ANY_CS  
BLOB  
CLOB CHARACTER SET ANY_CS  
BFILE
```

## プラグマ

```
pragma restrict_references(variable_value,RNDS,WNDS);
```

VARIABLE\_VALUE プロシージャでは、次の構文もサポートされます。

```
DBMS_SQL.VARIABLE_VALUE (  
    c                IN   INTEGER,  
    name             IN   VARCHAR2,  
    <table_variable> IN   <datatype>);
```

<table\_variable> とそれに対応する <datatype> は、次のいずれかの組合せが可能です。

```
<num_tab>      Number_Table  
<vchr2_tab>    Varchar2_Table  
<date_tab>     Date_Table  
<blob_tab>     Blob_Table  
<clob_tab>     Clob_Table  
<bfile_tab>    Bfile_Table
```

CHAR、RAW および ROWID データを含んだ変数では、次のバリエーションを構文で使用できます。

```
DBMS_SQL.VARIABLE_VALUE_CHAR (  
    c                IN   INTEGER,  
    name             IN   VARCHAR2,  
    value            OUT CHAR CHARACTER SET ANY_CS);
```

```
DBMS_SQL.VARIABLE_VALUE_RAW (  
    c                IN   INTEGER,  
    name             IN   VARCHAR2,  
    value            OUT RAW);
```

```
DBMS_SQL.VARIABLE_VALUE_ROWID (  
    c                IN   INTEGER,  
    name             IN   VARCHAR2,  
    value            OUT ROWID);
```

## パラメータ

表 48-13 VARIABLE\_VALUE プロシージャのパラメータ

パラメータ	説明
c	値を取得するカーソルの ID 番号。
name	値を検索する変数名。
value	指定した位置の変数の値を戻します。  この出力パラメータの型が、BIND_VARIABLE へのコールで定義されている値の実際の型と異なる場合、例外エラー ORA-06562、inconsistent_type が発生します。
position	カーソル内の列の相対位置。  文の最初の列は位置 1 です。

## 更新、挿入、削除の処理

動的 SQL を使用して INSERT、UPDATE または DELETE を処理する場合は、次のステップを実行する必要があります。

- 最初に、EXECUTE をコールして、INSERT 文、UPDATE 文または DELETE 文を実行します。
- 文に returning 句がある場合は、VARIABLE\_VALUE をコールして出力変数に割り当てられた値を取り出します。

## IS\_OPEN ファンクション

このファンクションは、指定のカーソルが現在オープンしているかどうかをチェックします。

### 構文

```
DBMS_SQL.IS_OPEN (  
    c                IN INTEGER)  
    RETURN BOOLEAN;
```

### プラグマ

```
pragma restrict_references(is_open,RNDS,WNDS);
```

パラメータ

表 48-14 IS\_OPEN ファンクションのパラメータ

パラメータ	説明
c	チェックするカーソルのカーソル ID 番号。

戻り値

表 48-15 IS\_OPEN ファンクションの戻り値

戻り値	説明
TRUE	指定のカーソルは、現在オープンしています。
FALSE	指定のカーソルは、現在オープンしていません。

DESCRIBE\_COLUMNS プロシージャ

このプロシージャは、DBMS\_SQL によってオープンして解析されたカーソルの列を記述します。

DESC\_REC 型

DBMS\_SQL パッケージでは、DESC\_REC レコード型を次のように宣言します。

```
type desc_rec is record (  
    col_type          BINARY_INTEGER := 0,  
    col_max_len       BINARY_INTEGER := 0,  
    col_name          VARCHAR2(32)   := '',  
    col_name_len      BINARY_INTEGER := 0,  
    col_schema_name   VARCHAR2(32)   := '',  
    col_schema_name_len BINARY_INTEGER := 0,  
    col_precision     BINARY_INTEGER := 0,  
    col_scale         BINARY_INTEGER := 0,  
    col_charsetid     BINARY_INTEGER := 0,  
    col_charsetform   BINARY_INTEGER := 0,  
    col_null_ok       BOOLEAN        := TRUE);
```

## パラメータ

表 48-16 DESC\_REC 型のパラメータ

パラメータ	説明
col_type	記述される列の型。
col_max_len	列の最大長。
col_name	列名。
col_name_len	列名の長さ。
col_schema_name	列の型が定義されたスキーマ名（オブジェクト型の場合）。
col_schema_name_len	スキーマの長さ。
col_precision	数値の場合は、列の精度。
col_scale	数値の場合は、列の位取り。
col_charsetid	列のキャラクタ・セットの識別子。
col_charsetform	列のキャラクタ・セット・フォーム。
col_null_ok	列で NULL が可能な場合は、TRUE。

## DESC\_TAB 型

DESC\_TAB 型は、DESC\_REC レコードの PL/SQL 表です。

type desc\_tab is table of desc\_rec index by BINARY\_INTEGER;

ローカル変数を PL/SQL 表型である DESC\_TAB として宣言し、次に、DESCRIBE\_COLUMNS プロシージャをコールして、各列の説明を表に入れることができます。その際、すべての列が記述され、単一の列のみを記述することはできません。

## 構文

```
DBMS_SQL.DESCRIBE_COLUMNS (  
    c                IN  INTEGER,  
    col_cnt          OUT INTEGER,  
    desc_t           OUT DESC_TAB);
```

パラメータ

表 48-17 DBMS\_SQL.DESCRIBE\_COLUMNS プロシージャのパラメータ

パラメータ	説明
c	記述される列のカーソルの ID 番号。
col_cnt	問合せの選択リストにある列数。
desc_t	DESC_REC の表。各 DESC_REC が問合せ内の列を説明します。

関連項目： DESCRIBE\_COLUMNS の使用方法は、「例 8: 列の記述」( 48-39 ページ) を参照してください。

CLOSE\_CURSOR プロシージャ

このファンクションは、指定のカーソルをクローズします。

構文

```
DBMS_SQL.CLOSE_CURSOR (  
    c      IN OUT INTEGER);
```

プラグマ

```
pragma restrict_references(close_cursor,RNDS,WNDS);
```

パラメータ

表 48-18 CLOSE\_CURSOR プロシージャのパラメータ

パラメータ	モード	説明
c	IN	クローズするカーソルの ID 番号。
c	OUT	カーソルは NULL に設定されています。  CLOSE_CURSOR をコールした後、カーソルに割り当てられたメモリーは解放され、そのカーソルからはフェッチできなくなります。



## エラーの位置

DBMS\_SQL パッケージには、セッションで最後に参照されたカーソルの情報を取得するための追加ファンクションがいくつかあります。これらのファンクションが戻す値は、SQL 文の実行直後にだけ意味を持ちます。また、エラーを検出するファンクションは、特定の DBMS\_SQL コール後にだけ意味を持ちます。たとえば、PARSE 直後に LAST\_ERROR\_POSITION をコールします。

## LAST\_ERROR\_POSITION ファンクション

このファンクションは、エラーが発生した SQL 文テキスト内のバイト・オフセットを戻します。SQL 文内の最初の文字は、位置 0 にあります。

### 構文

```
DBMS_SQL.LAST_ERROR_POSITION  
    RETURN INTEGER;
```

### パラメータ

なし。

### プラグマ

```
pragma restrict_references(last_error_position,RNDS,WNDS);
```

### 使用上の注意

このファンクションは、別の DBMS\_SQL プロシージャまたはファンクションのコール前、かつ PARSE のコール後にコールしてください。

## LAST\_ROW\_COUNT ファンクション

このファンクションは、フェッチされた累積行数を戻します。

### 構文

```
DBMS_SQL.LAST_ROW_COUNT  
    RETURN INTEGER;
```

### パラメータ

なし。

### プラグマ

```
pragma restrict_references(last_row_count,RNDS,WNDS);
```

### 使用上の注意

このファンクションは、`FETCH_ROWS` コールまたは `EXECUTE_AND_FETCH` コール後にコールしてください。`EXECUTE` コール後にコールすると、戻される値はゼロです。

## LAST\_ROW\_ID ファンクション

このファンクションは、処理された最後の行の `ROWID` を戻します。

### 構文

```
DBMS_SQL.LAST_ROW_ID  
    RETURN ROWID;
```

### パラメータ

なし。

### プラグマ

```
pragma restrict_references(last_row_id,RNDS,WNDS);
```

### 使用上の注意

このファンクションは、`FETCH_ROWS` コールまたは `EXECUTE_AND_FETCH` コール後にコールしてください。

## LAST\_SQL\_FUNCTION\_CODE ファンクション

このファンクションは、文の `SQL` ファンクション・コードを戻します。これらのコードは、『Oracle8i コール・インタフェース・プログラマーズ・ガイド』に一覧があります。

### 構文

```
DBMS_SQL.LAST_SQL_FUNCTION_CODE  
    RETURN INTEGER;
```

### パラメータ

なし。

### プラグマ

```
pragma restrict_references(last_sql_function_code,RNDS,WNDS);
```

## 使用上の注意

このファンクションは、SQL 文の実行直後にコールする必要があります。そうでない場合は、戻り値が未定義になります。

## 例

このセクションには、DBMS\_SQL パッケージを使用するプロシージャの例が記述されています。

**例 1:** 次のプロシージャの例は、プロシージャを SQL 文に渡し、その SQL 文を解析して実行します。

```
CREATE OR REPLACE PROCEDURE exec(string IN varchar2) AS
    cursor_name INTEGER;
    ret INTEGER;
BEGIN
    cursor_name := DBMS_SQL.OPEN_CURSOR;
```

DDL 文は PARSE をコールして実行され、暗黙のコミットが実行されます。

```
    DBMS_SQL.PARSE(cursor_name, string, DBMS_SQL.native);
    ret := DBMS_SQL.EXECUTE(cursor_name);
    DBMS_SQL.CLOSE_CURSOR(cursor_name);
END;
```

このようなプロシージャを作成すると、次の操作を実行できます。

- コール側のプログラムによって、実行時に SQL 文を動的に生成できます。
- SQL 文は、DDL 文またはバインドなしの DML で構いません。

たとえば、このプロシージャの作成後に、次のコールを行うことができます。

```
exec('create table acct(c1 integer)');
```

次の例のように、このプロシージャはリモートでコールすることもできます。これによって、リモート DDL を実行できます。

```
exec@hq.com('CREATE TABLE acct(c1 INTEGER)');
```

**例 2:** 次のプロシージャの例は、ソースと宛先表の名前が渡され、ソース表から宛先表に行をコピーします。このプロシージャの例は、ソース表と宛先表にはいずれも次の列があることを前提としています。

```
id          of type NUMBER
name        of type VARCHAR2(30)
birthdate  of type DATE
```

このプロシージャでは、動的 SQL を使用する必要は特にありませんが、ここでは、このパッケージの概念をわかりやすく説明しています。

```
CREATE OR REPLACE PROCEDURE copy (
    source      IN VARCHAR2,
    destination IN VARCHAR2) IS
    id_var      NUMBER;
    name_var    VARCHAR2(30);
    birthdate_var DATE;
    source_cursor INTEGER;
    destination_cursor INTEGER;
    ignore      INTEGER;
BEGIN

-- Prepare a cursor to select from the source table:
source_cursor := dbms_sql.open_cursor;
DBMS_SQL.PARSE(source_cursor,
    'SELECT id, name, birthdate FROM ' || source,
    DBMS_SQL.native);
DBMS_SQL.DEFINE_COLUMN(source_cursor, 1, id_var);
DBMS_SQL.DEFINE_COLUMN(source_cursor, 2, name_var, 30);
DBMS_SQL.DEFINE_COLUMN(source_cursor, 3, birthdate_var);
ignore := DBMS_SQL.EXECUTE(source_cursor);

-- Prepare a cursor to insert into the destination table:
destination_cursor := DBMS_SQL.OPEN_CURSOR;
DBMS_SQL.PARSE(destination_cursor,
    'INSERT INTO ' || destination ||
    ' VALUES (:id_bind, :name_bind, :birthdate_bind)',
    DBMS_SQL.native);

-- Fetch a row from the source table and insert it into the destination table:
LOOP
    IF DBMS_SQL.FETCH_ROWS(source_cursor)>0 THEN
        -- get column values of the row
        DBMS_SQL.COLUMN_VALUE(source_cursor, 1, id_var);
        DBMS_SQL.COLUMN_VALUE(source_cursor, 2, name_var);
        DBMS_SQL.COLUMN_VALUE(source_cursor, 3, birthdate_var);

-- Bind the row into the cursor that inserts into the destination table. You
-- could alter this example to require the use of dynamic SQL by inserting an
-- if condition before the bind.
        DBMS_SQL.BIND_VARIABLE(destination_cursor, ':id_bind', id_var);
        DBMS_SQL.BIND_VARIABLE(destination_cursor, ':name_bind', name_var);
        DBMS_SQL.BIND_VARIABLE(destination_cursor, ':birthdate_bind',
birthdate_var);
        ignore := DBMS_SQL.EXECUTE(destination_cursor);
```

```

ELSE

-- No more rows to copy:
    EXIT;
    END IF;
END LOOP;

-- Commit and close all cursors:
COMMIT;
DBMS_SQL.CLOSE_CURSOR(source_cursor);
DBMS_SQL.CLOSE_CURSOR(destination_cursor);
EXCEPTION
    WHEN OTHERS THEN
        IF DBMS_SQL.IS_OPEN(source_cursor) THEN
            DBMS_SQL.CLOSE_CURSOR(source_cursor);
        END IF;
        IF DBMS_SQL.IS_OPEN(destination_cursor) THEN
            DBMS_SQL.CLOSE_CURSOR(destination_cursor);
        END IF;
        RAISE;
END;
/

```

**例 3、4 および 5: 一括 DML 次の一連の例では、DELETE、INSERT および UPDATE の各 SQL DML 文での一括配列バインド（表項目）の使用方法を示します。**

たとえば、DELETE 文では、WHERE 句に配列をバインドし、配列内の要素ごとに文を実行できます。

```

declare
    stmt varchar2(200);
    dept_no_array dbms_sql.Number_Table;
    c number;
    dummy number;
begin
    dept_no_array(1) := 10; dept_no_array(2) := 20;
    dept_no_array(3) := 30; dept_no_array(4) := 40;
    dept_no_array(5) := 30; dept_no_array(6) := 40;
    stmt := 'delete from emp where deptno = :dept_array';
    c := dbms_sql.open_cursor;
    dbms_sql.parse(c, stmt, dbms_sql.native);
    dbms_sql.bind_array(c, ':dept_array', dept_no_array, 1, 4);
    dummy := dbms_sql.execute(c);
    dbms_sql.close_cursor(c);

    exception when others then
        if dbms_sql.is_open(c) then

```

```
        dbms_sql.close_cursor(c);
    end if;
    raise;
end;
/
```

前述の例では、1 から 4 までの要素のみが、bind\_array コールで指定したとおりに使用されます。配列の各要素は、大量の従業員をデータベースから削除する可能性があります。

次に、一括 INSERT 文の例を示します。

```
declare
    stmt varchar2(200);
    empno_array dbms_sql.Number_Table;
    empname_array dbms_sql.Varchar2_Table;
    c number;
    dummy number;
begin
    for i in 0..9 loop
        empno_array(i) := 1000 + i;
        empname_array(i) := get_name(i);
    end loop;
    stmt := 'insert into emp values(:num_array, :name_array)';
    c := dbms_sql.open_cursor;
    dbms_sql.parse(c, stmt, dbms_sql.native);
    dbms_sql.bind_array(c, ':num_array', empno_array);
    dbms_sql.bind_array(c, ':name_array', empname_array);
    dummy := dbms_sql.execute(c);
    dbms_sql.close_cursor(c);

    exception when others then
        if dbms_sql.is_open(c) then
            dbms_sql.close_cursor(c);
        end if;
        raise;
end;
/
```

実行が開始されると、10 人の従業員はすべて表に挿入されます。

最後に、一括 UPDATE 文の例を示します。

```
declare
    stmt varchar2(200);
    emp_no_array dbms_sql.Number_Table;
    emp_addr_array dbms_sql.Varchar2_Table;
    c number;
    dummy number;
begin
```

```

for i in 0..9 loop
    emp_no_array(i) := 1000 + i;
    emp_addr_array(i) := get_new_addr(i);
end loop;
stmt := 'update emp set ename = :name_array
        where empno = :num_array';
c := dbms_sql.open_cursor;
dbms_sql.parse(c, stmt, dbms_sql.native);
dbms_sql.bind_array(c, ':num_array', empno_array);
dbms_sql.bind_array(c, ':name_array', empname_array);
dummy := dbms_sql.execute(c);
dbms_sql.close_cursor(c);

exception when others then
    if dbms_sql.is_open(c) then
        dbms_sql.close_cursor(c);
    end if;
    raise;
end;
/

```

EXECUTE がコールされると、全従業員のアドレスが一度に更新されます。2つのコレクションの処理は、いつも同時に進行します。WHERE 句で複数の行が戻される場合、全従業員は、その時点で addr\_array が参照するアドレスを取得します。

**例 6 および 7: 配列の定義** 次の例では、DEFINE\_ARRAY プロシージャの使用方法を示します。

```

declare
    c      number;
    d      number;
    n_tab  dbms_sql.Number_Table;
    indx   number := -10;
begin
    c := dbms_sql.open_cursor;
    dbms_sql.parse(c, 'select n from t order by 1', dbms_sql);

    dbms_sql.define_array(c, 1, n_tab, 10, indx);

    d := dbms_sql.execute(c);
    loop
        d := dbms_sql.fetch_rows(c);

        dbms_sql.column_value(c, 1, n_tab);

        exit when d != 10;
    end loop;

```

```

dbms_sql.close_cursor(c);

exception when others then
  if dbms_sql.is_open(c) then
    dbms_sql.close_cursor(c);
  end if;
  raise;
end;
/

```

前述の例では `FETCH_ROWS` をコールするたびに、`DBMS_SQL` バッファに保持されている 10 行がフェッチされます。`COLUMN_VALUE` コールが実行されると、それらの行は指定した PL/SQL 表（この場合は `n_tab`）の `DEFINE` 文で指定した -10 から -1 の位置に移動します。次に、2 番目のバッチがループ内でフェッチされ、行が 0 から 9 の位置に移動し、あとは同様に続きます。

各配列への現行の索引は、自動的にメンテナンスされます。この索引は、`EXECUTE` 時に "indx" に初期化され、`COLUMN_VALUE` がコールされるたびに更新されます。任意の時点で再実行した場合、各 `DEFINE` の現行の索引は "indx" に再初期化されます。

このようにして、問合せのすべての結果が表内にフェッチされます。`FETCH_ROWS` で 10 行をフェッチできない場合は、実際にフェッチされた行数を戻して（1 行もフェッチできなかった場合はゼロを戻します）ループを終了します。

`DEFINE_ARRAY` プロシージャの別の使用例を次に示します。

次のように定義された `MULTI_TAB` 表を想定します。

```

create table multi_tab (num number,
                        dat1 date,
                        var varchar2(24),
                        dat2 date)

```

この表からすべてを選択して 4 つの PL/SQL 表に移動するには、次の簡単なプログラムを使用できます。

```

declare
  c      number;
  d      number;
  n_tab  dbms_sql.Number_Table;
  d_tab1 dbms_sql.Date_Table;
  v_tab  dbms_sql.Varchar2_Table;
  d_tab2 dbms_sql.Date_Table;
  indx number := 10;
begin

  c := dbms_sql.open_cursor;
  dbms_sql.parse(c, 'select * from multi_tab order by 1', dbms_sql);

```



```

dbms_sql.define_array(c, 1, n_tab, 5, indx);
dbms_sql.define_array(c, 2, d_tab1, 5, indx);
dbms_sql.define_array(c, 3, v_tab, 5, indx);
dbms_sql.define_array(c, 4, d_tab2, 5, indx);

d := dbms_sql.execute(c);

loop
    d := dbms_sql.fetch_rows(c);

    dbms_sql.column_value(c, 1, n_tab);
    dbms_sql.column_value(c, 2, d_tab1);
    dbms_sql.column_value(c, 3, v_tab);
    dbms_sql.column_value(c, 4, d_tab2);

    exit when d != 5;
end loop;

dbms_sql.close_cursor(c);

/*

```

これで、4つの表はあらゆる用途に使用できます。使用方法の1つは、'INSERT into SOME\_T values (:a, :b, :c, :d)' などの問合せを使用して、行を他の表に移動するために BIND\_ARRAY を使用できます。

```

*/

exception when others then
    if dbms_sql.is_open(c) then
        dbms_sql.close_cursor(c);
    end if;
    raise;
end;
/

```

**例 8: 列の記述** この例は、記述する表に対して SELECT \* による問合せを使用し、SQL\*Plus の DESCRIBE コールのかわりに使用できます。

```

declare
    c number;
    d number;
    col_cnt integer;
    f boolean;
    rec_tab dbms_sql.desc_tab;
    col_num number;

```

```

procedure print_rec(rec in dbms_sql.desc_rec) is
begin
  dbms_output.new_line;
  dbms_output.put_line('col_type           = '
                        || rec.col_type);
  dbms_output.put_line('col_maxlen        = '
                        || rec.col_max_len);
  dbms_output.put_line('col_name          = '
                        || rec.col_name);
  dbms_output.put_line('col_name_len      = '
                        || rec.col_name_len);
  dbms_output.put_line('col_schema_name   = '
                        || rec.col_schema_name);
  dbms_output.put_line('col_schema_name_len = '
                        || rec.col_schema_name_len);
  dbms_output.put_line('col_precision    = '
                        || rec.col_precision);
  dbms_output.put_line('col_scale        = '
                        || rec.col_scale);
  dbms_output.put('col_null_ok          = ');
  if (rec.col_null_ok) then
    dbms_output.put_line('true');
  else
    dbms_output.put_line('false');
  end if;
end;

begin
  c := dbms_sql.open_cursor;

  dbms_sql.parse(c, 'select * from scott.bonus', dbms_sql);

  d := dbms_sql.execute(c);

  dbms_sql.describe_columns(c, col_cnt, rec_tab);

/*
 * Following loop could simply be for j in 1..col_cnt loop.
 * Here we are simply illustrating some of the PL/SQL table
 * features.
 */
  col_num := rec_tab.first;
  if (col_num is not null) then
    loop
      print_rec(rec_tab(col_num));
      col_num := rec_tab.next(col_num);
      exit when (col_num is null);
    end loop;
  end if;
end;

```

```

end if;

dbms_sql.close_cursor(c);
end;
/

```

**例 9: RETURNING 句** RETURNING 句は、Oracle 8.0.3 では DML 文に追加されていました。この句を使用して、INSERT 文、UPDATE 文および DELETE 文は、式の値を戻すことができます。この値は、バインド変数に戻されます。

単一行が挿入、更新または削除される場合は、DBMS\_SQL.BIND\_VARIABLE を使用して、これらのアウトバインドをバインドします。複数の行が挿入、更新または削除された場合は、DBMS\_SQL.BIND\_ARRAY を使用します。これらのバインド変数の値を取得するには、DBMS\_SQL.VARIABLE\_VALUE をコールする必要があります。

---

**注意：** これは、DBMS\_SQL 内でアウトバインドを使用した PL/SQL ブロックを実行した後で、DBMS\_SQL.VARIABLE\_VALUE をコールする必要があることに似ています。

---

#### i) 単一行の挿入

```

create or replace procedure single_Row_insert
(c1 number, c2 number, r out number) is
c number;
n number;
begin
c := dbms_sql.open_cursor;
dbms_sql.parse(c, 'insert into tab values (:bnd1, :bnd2)' ||
'returning c1*c2 into :bnd3', 2);
dbms_sql.bind_variable(c, 'bnd1', c1);
dbms_sql.bind_variable(c, 'bnd2', c2);
dbms_sql.bind_variable(c, 'bnd3', r);
n := dbms_sql.execute(c);
dbms_sql.variable_value(c, 'bnd3', r); -- get value of outbind variable
dbms_sql.close_cursor(c);
end;
/

```

#### ii) 単一行の更新

```

create or replace procedure single_Row_update
(c1 number, c2 number, r out number) is
c number;
n number;
begin
c := dbms_sql.open_cursor;

```

```

dbms_sql.parse(c, 'update tab set c1 = :bnd1, c2 = :bnd2 ' ||
                  'where rownum < 2' ||
                  'returning c1*c2 into :bnd3', 2);
dbms_sql.bind_variable(c, 'bnd1', c1);
dbms_sql.bind_variable(c, 'bnd2', c2);
dbms_sql.bind_variable(c, 'bnd3', r);
n := dbms_sql.execute(c);
dbms_sql.variable_value(c, 'bnd3', r);-- get value of outbind variable
dbms_sql.close_cursor(c);
end;
/

```

### iii) 単一行の削除

```

create or replace procedure single_Row_Delete
(c1 number, c2 number, r out number) is
c number;
n number;
begin
c := dbms_sql.open_cursor;
dbms_sql.parse(c, 'delete from tab ' ||
                  'where rownum < 2' ||
                  'returning c1*c2 into :bnd3', 2);
dbms_sql.bind_variable(c, 'bnd1', c1);
dbms_sql.bind_variable(c, 'bnd2', c2);
dbms_sql.bind_variable(c, 'bnd3', r);
n := dbms_sql.execute(c);
dbms_sql.variable_value(c, 'bnd3', r);-- get value of outbind variable
dbms_sql.close_cursor(c);
end;
/

```

### iv) 複数行の挿入

```

create or replace procedure multi_Row_insert
(c1 dbms_sql.number_table, c2 dbms_sql.number_table,
r out dbms_sql.number_table) is
c number;
n number;
begin
c := dbms_sql.open_cursor;
dbms_sql.parse(c, 'insert into tab values (:bnd1, :bnd2) ' ||
                  'returning c1*c2 into :bnd3', 2);
dbms_sql.bind_array(c, 'bnd1', c1);
dbms_sql.bind_array(c, 'bnd2', c2);
dbms_sql.bind_array(c, 'bnd3', r);
n := dbms_sql.execute(c);
dbms_sql.variable_value(c, 'bnd3', r);-- get value of outbind variable

```

```

        dbms_Sql.close_Cursor(c);
    end;
/

```

#### v) 複数行の更新

```

create or replace procedure multi_Row_update
    (c1 number, c2 number, r out dbms_Sql.number_table) is
    c number;
    n number;
begin
    c := dbms_sql.open_cursor;
    dbms_sql.parse(c, 'update tab set c1 = :bnd1 where c2 = :bnd2 ' ||
        'returning c1*c2 into :bnd3', 2);
    dbms_sql.bind_variable(c, 'bnd1', c1);
    dbms_sql.bind_variable(c, 'bnd2', c2);
    dbms_sql.bind_array(c, 'bnd3', r);
    n := dbms_sql.execute(c);
    dbms_sql.variable_value(c, 'bnd3', r);-- get value of outbind variable
    dbms_Sql.close_Cursor(c);
end;
/

```

---

**注意：** bnd1 と bnd2 は、同様に配列にできます。更新されたすべての行に対する式の値は、bnd3 に入れます。bnd1 と bnd2 の各値について、どの行が更新されたかを区別する方法はありません。

---

#### vi) 複数行の削除

```

create or replace procedure multi_row_delete
    (c1 dbms_Sql.number_table,
    r out dbms_sql.number_table) is
    c number;
    n number;
begin
    c := dbms_sql.open_cursor;
    dbms_sql.parse(c, 'delete from tab where c1 = :bnd1' ||
        'returning c1*c2 into :bnd2', 2);
    dbms_sql.bind_array(c, 'bnd1', c1);
    dbms_sql.bind_array(c, 'bnd2', r);
    n := dbms_sql.execute(c);
    dbms_sql.variable_value(c, 'bnd2', r);-- get value of outbind variable
    dbms_Sql.close_Cursor(c);
end;
/

```

vii) 一括 PL/SQL でのアウトバインド

```
create or replace foo (n number, square out number) is
begin square := n * n; end;/

create or replace procedure bulk_plsql
  (n dbms_sql.number_Table, square out dbms_sql.number_Table) is
c number;
r number;
begin
  c := dbms_sql.open_cursor;
  dbms_sql.parse(c, 'begin foo(:bnd1, :bnd2); end;', 2);
  dbms_sql.bind_array(c, 'bnd1', n);
  dbms_sql.bind_array(c, 'bnd2', square);
  r := dbms_sql.execute(c);
  dbms_sql.variable_value(c, 'bnd2', square);
end;
/
```

---

---

**注意：** number\_Table の DBMS\_SQL.BIND\_ARRAY は、数値を内部的にバインドします。文を実行する回数は、インバインド配列内の要素数によって決まります。

---

---

DBMS\_STATS は、データベース・オブジェクトについて収集されたオプティマイザの統計情報を表示および変更するためのメカニズムを提供します。統計情報は、次の 2 つの場所に常駐させることができます。

1. ディクショナリ。
2. この目的のためにユーザーのスキーマ内に作成された表。

コストベースのオプティマイザに影響を与える統計情報は、ディクショナリに格納されている統計情報のみです。

また、このパッケージによって各種の統計情報を並列的に簡単に収集できます。パッケージは、次の 3 つの主なセクションに分割されています。

- [統計情報の設定または取得](#)
- [統計情報の転送](#)
- [オプティマイザ統計情報の収集](#)

---

## DBMS\_STATS の使用方法

ほとんどの DBMS\_STATS プロシージャには、statown、stattab および statid の 3 つのパラメータが含まれています。これらのパラメータによって、ユーザーはオプティマイザに影響を与えない自分自身の表（ディクショナリ外）に統計情報を格納できます。したがって、ユーザーは統計情報のセットをメンテナンスおよび試用することができます。

stattab パラメータで、統計情報を保持する表の名前を指定します。この表は、（statown パラメータが指定されていない限り）統計情報が収集されるオブジェクトと同じスキーマ内に常駐しているとみなされます。ユーザーは、異なる stattab 識別子で複数の表を作成し、統計情報セットを別々に保持できます。

さらに、statid パラメータを使用して 1 つの stattab 内で複数の統計情報セットをメンテナンスできるため、ユーザーのスキーマが混乱するのを回避できます。

すべての SET または GET プロシージャについて、stattab が準備されていない場合（つまり NULL の場合）操作はディクショナリにある統計情報に対して直接行われます。したがって、ディクショナリを直接変更するのみの場合は、統計表の作成は不要です。ただし、stattab が NULL でない場合、SET または GET 操作は、指定したユーザー統計表に対して行われ、ディクショナリに対しては行われません。

## 型

最小 / 最大値とヒストグラム終点の型は、次のとおりです。

```
TYPE numarray IS VARRAY(256) OF NUMBER;
TYPE datearray IS VARRAY(256) OF DATE;
TYPE chararray IS VARRAY(256) OF VARCHAR2(4000);
TYPE rawarray IS VARRAY(256) OF RAW(2000);
```

```
type StatRec is record (
    epc      NUMBER,
    minval   RAW(2000),
    maxval   RAW(2000),
    bkvals   NUMARRAY,
    novals   NUMARRAY);
```

失効した表のリストの型は、次のとおりです。

```
type ObjectElem is record (
    ownname   VARCHAR2(30),      -- owner
    objtype   VARCHAR2(6),      -- 'TABLE' or 'INDEX'
    objname   VARCHAR2(30),      -- table/index
    partname   VARCHAR2(30),      -- partition
    subpartname VARCHAR2(30),      -- subpartition
    confidence NUMBER);          -- not used
type ObjectTab is TABLE of ObjectElem;
```



## サブプログラムの要約

表 49-1 DBMS\_STATS パッケージのサブプログラム

サブプログラム	説明
<a href="#">PREPARE_COLUMN_VALUES プロシージャ</a> (49-5 ページ)	ユーザー指定の最小値、最大値およびヒストグラム終点のデータ型固有値を、SET_COLUMN_STATS を介して将来格納するために Oracle の内部表記に変換します。
<a href="#">SET_COLUMN_STATS プロシージャ</a> (49-7 ページ)	列に関連する情報を設定します。
<a href="#">SET_INDEX_STATS プロシージャ</a> (49-9 ページ)	索引に関連する情報を設定します。
<a href="#">SET_TABLE_STATS プロシージャ</a> (49-10 ページ)	表に関連する情報を設定します。
<a href="#">CONVERT_RAW_VALUE プロシージャ</a> (49-11 ページ)	最大値または最小値の内部表記を、データ型固有値に変換します。
<a href="#">GET_COLUMN_STATS プロシージャ</a> (49-13 ページ)	列に関連するすべての情報を取得します。
<a href="#">GET_INDEX_STATS プロシージャ</a> (49-14 ページ)	索引に関連するすべての情報を取得します。
<a href="#">GET_TABLE_STATS プロシージャ</a> (49-15 ページ)	表に関連するすべての情報を取得します。
<a href="#">DELETE_COLUMN_STATS プロシージャ</a> (49-16 ページ)	列に関連する統計情報を削除します。
<a href="#">DELETE_INDEX_STATS プロシージャ</a> (49-17 ページ)	索引に関連する統計情報を削除します。
<a href="#">DELETE_TABLE_STATS プロシージャ</a> (49-18 ページ)	表に関連する統計情報を削除します。
<a href="#">DELETE_SCHEMA_STATS プロシージャ</a> (49-20 ページ)	スキーマに関連する統計情報を削除します。
<a href="#">DELETE_DATABASE_STATS プロシージャ</a> (49-20 ページ)	データベース全体に関する統計情報を削除します。
<a href="#">CREATE_STAT_TABLE プロシージャ</a> (49-22 ページ)	統計情報を保持できる ownname のスキーマにstattab の名前で表を作成します。
<a href="#">DROP_STAT_TABLE プロシージャ</a> (49-22 ページ)	CREATE_STAT_TABLE で作成したユーザー統計表を削除します。

表 49-1 DBMS\_STATS パッケージのサブプログラム

サブプログラム	説明
<a href="#">EXPORT_COLUMN_STATS プロシージャ</a> (49-23 ページ)	特定の列に関する統計情報を取り出し、stattab で識別されるユーザー統計表に格納します。
<a href="#">EXPORT_INDEX_STATS プロシージャ</a> (49-24 ページ)	特定の索引に関する統計情報を取り出し、stattab で識別されるユーザー統計表に格納します。
<a href="#">EXPORT_TABLE_STATS プロシージャ</a> (49-25 ページ)	特定の表に関する統計情報を取り出し、ユーザー統計表に格納します。
<a href="#">EXPORT_SCHEMA_STATS プロシージャ</a> (49-26 ページ)	ownname で識別されるスキーマ内のすべてのオブジェクトに関する統計情報を取り出し、stattab で識別されるユーザー統計表に格納します。
<a href="#">EXPORT_DATABASE_STATS プロシージャ</a> (49-26 ページ)	データベース内のすべてのオブジェクトに関する統計情報を取り出し、statown.stattab で識別されるユーザー統計表に格納します。
<a href="#">IMPORT_COLUMN_STATS プロシージャ</a> (49-27 ページ)	stattab で識別されるユーザー統計表から特定の列に関する統計情報を取り出し、ディクショナリに格納します。
<a href="#">IMPORT_INDEX_STATS プロシージャ</a> (49-28 ページ)	stattab で識別されるユーザー統計表から特定の索引に関する統計情報を取り出し、ディクショナリに格納します。
<a href="#">IMPORT_TABLE_STATS プロシージャ</a> (49-29 ページ)	stattab で識別されるユーザー統計表から特定の表に関する統計情報を取り出し、ディクショナリに格納します。
<a href="#">IMPORT_SCHEMA_STATS プロシージャ</a> (49-30 ページ)	ownname で識別されるスキーマ内のすべてのオブジェクトに関する統計情報をユーザー統計表から取り出し、ディクショナリに格納します。
<a href="#">IMPORT_DATABASE_STATS プロシージャ</a> (49-30 ページ)	データベース内のすべてのオブジェクトに関する統計情報をユーザー統計表から取り出し、ディクショナリに格納します。
<a href="#">GATHER_INDEX_STATS プロシージャ</a> (49-31 ページ)	索引の統計情報を収集します。
<a href="#">GATHER_TABLE_STATS プロシージャ</a> (49-32 ページ)	表と列（および索引）の統計情報を収集します。
<a href="#">GATHER_SCHEMA_STATS プロシージャ</a> (49-34 ページ)	スキーマ内のすべてのオブジェクトに関する統計情報を収集します。
<a href="#">GATHER_DATABASE_STATS プロシージャ</a> (49-37 ページ)	データベース内のすべてのオブジェクトに関する統計情報を収集します。
<a href="#">GENERATE_STATS プロシージャ</a> (49-39 ページ)	関連するオブジェクトに関して以前に収集した統計情報から、オブジェクトの統計情報を生成します。

## 統計情報の設定または取得

次のプロシージャによって、列、索引および表に関連する個別の統計情報を格納したり、取り出すことができます。

```
PREPARE_COLUMN_VALUES
SET_COLUMN_STATS
SET_INDEX_STATS
SET_TABLE_STATS
```

```
CONVERT_RAW_VALUE
GET_COLUMN_STATS
GET_INDEX_STATS
GET_TABLE_STATS
```

```
DELETE_COLUMN_STATS
DELETE_INDEX_STATS
DELETE_TABLE_STATS
DELETE_SCHEMA_STATS
DELETE_DATABASE_STATS
```

## PREPARE\_COLUMN\_VALUES プロシージャ

このプロシージャは、ユーザー指定の最小値、最大値およびヒストグラム終点のデータ型固有値を、SET\_COLUMN\_STATS を介して将来格納するために Oracle の内部表記に変換します。

### 構文

```
DBMS_STATS.PREPARE_COLUMN_VALUES (
    srec      IN OUT StatRec,
    charvals  CHARARRAY);

DBMS_STATS.PREPARE_COLUMN_VALUES (
    srec      IN OUT StatRec,
    datevals  DATEARRAY);

DBMS_STATS.PREPARE_COLUMN_VALUES (
    srec      IN OUT StatRec,
    numvals   NUMARRAY);

DBMS_STATS.PREPARE_COLUMN_VALUES (
    srec      IN OUT StatRec,
    rawvals   RAWARRAY);

DBMS_STATS.PREPARE_COLUMN_VALUES_NVARCHAR (
    srec      IN OUT StatRec,
```

```
nvmin      NVARCHAR2,  
nvmax      NVARCHAR2);  
  
DBMS_STATS.PREPARE_COLUMN_VALUES_ROWID (  
  srec  IN OUT StatRec,  
  rwmin      ROWID,  
  rwmax      ROWID);
```

プラグマ

```
pragma restrict_references(prepare_column_values, WNDS, RNDS, WNPS, RNPS);  
pragma restrict_references(prepare_column_values_nvarchar, WNDS, RNDS, WNPS, RNPS);  
pragma restrict_references(prepare_column_values_rowid, WNDS, RNDS, WNPS, RNPS);
```

パラメータ

表 49-2 PREPARE\_COLUMN\_VALUES プロシージャのパラメータ

パラメータ	説明
srec.epc	charvals、datevals、numvals または rawvals で指定される値の数。この値は 2 ~ 256 の範囲で指定し、ヒストグラム情報を認めないプロシージャ（nvarchar と rowid）では 2 を設定する必要があります。  最初に対応する配列のエントリは列の最小値を含み、最後のエントリは最大値を含みます。2 つを超えるエントリがある場合、その他のエントリは、残りの高さ調整ヒストグラムまたは頻度ヒストグラムの終点値（小さい値から大きい値に順序立てられた中間値を持つ）を含みます。この値は圧縮のために調整できるため、戻り値は、SET_COLUMN_STATS へのコールに対して現状のままになります。
srec.bkvals	頻度分布が必要な場合、この配列には、charvals、datevals、numvals または rawvals で指定されている各個別値の発生回数が含まれています。そうでない場合、この配列は単なるアウトプット・パラメータであり、このプロシージャのコール時には NULL が設定されている必要があります。

データ型固有の入力パラメータ（次から 1 つを選択）。

charvals	列型が文字ベースの場合の値の配列。各文字列の最初の 32 バイトまでが使用されます。配列のエントリ数は、2 ~ 256 の範囲である必要があります。
datevals	列型が日付ベースの場合の値の配列。
numvals	列型が数値ベースの場合の値の配列。

rawvals	列型が RAW の場合の値の配列。各文字列の最初の 32 バイトまでが使用されます。
nvmin、nvmax	列型が各国文字キャラクタ・セット ( NLS ) の場合の最大値と最小値。この型の列について、ヒストグラム情報は提供できません。
rwmin、rwmax	列型が rowid の場合の最大値と最小値。この型の列について、ヒストグラム情報は提供できません。

アウトプット・パラメータ

表 49-3 PREPARE\_COLUMN\_VALUES プロシージャのアウトプット・パラメータ

パラメータ	説明
srec.minval	SET_COLUMN_STATS へのコールでの使用に適した最小値の内部表記。
srec.maxval	SET_COLUMN_STATS へのコールでの使用に適した最大値の内部表記。
srec.bkvals	SET_COLUMN_STATS へのコールでの使用に適した配列。
srec.novals	SET_COLUMN_STATS へのコールでの使用に適した配列。

例外

ORA-20001: 入力値が無効か、または矛盾しています。

SET\_COLUMN\_STATS プロシージャ

このプロシージャは、列に関連する情報を設定します。

構文

```
DBMS_STATS.SET_COLUMN_STATS (  
    ownname  VARCHAR2,  
    tabname  VARCHAR2,  
    colname  VARCHAR2,  
    partname VARCHAR2 DEFAULT NULL,  
    stattab  VARCHAR2 DEFAULT NULL,  
    statid   VARCHAR2 DEFAULT NULL,  
    statown  VARCHAR2 DEFAULT NULL,  
    distcnt  NUMBER    DEFAULT NULL,  
    density  NUMBER    DEFAULT NULL,  
    nullcnt  NUMBER    DEFAULT NULL,  
    srec     StatRec   DEFAULT NULL,  
    avgclen  NUMBER    DEFAULT NULL,
```

```
flags      NUMBER      DEFAULT NULL,
statown    VARCHAR2    DEFAULT NULL);
```

パラメータ

表 49-4 SET\_COLUMN\_STATS プロシージャのパラメータ

パラメータ	説明
ownname	スキーマ名。
tabname	この列が所属する表の名前。
colname	列名。
partname	統計情報を格納する表パーティションの名前。表がパーティション化されていて、partname が NULL の場合、統計情報はグローバルな表レベルで格納されます。
stattab	統計情報の格納場所を示すユーザー統計表の識別子。stattab が NULL の場合、統計情報はディクショナリ内に直接格納されます。
statid	stattab 内の統計情報を関連付ける識別子（オプション）（stattab が NULL でない場合のみ該当します）。
distcnt	個別値の数。
density	列密度。この値が NULL で、distcnt が NULL でない場合、密度は distcnt から導出されます。
nullcnt	NULL の数。
srec	PREPARE_COLUMN_VALUES または GET_COLUMN_STATS へのコールで入力された StatRec 構造。
avgcflen	列の平均長（単位はバイト）。
flags	Oracle 内部で使用（NULL のままにします）。
statown	stattab を含んだスキーマ（ownname と異なる場合）。

例外

- ORA-20000: オブジェクトが存在しないか、または権限が不十分です。
- ORA-20001: 入力値が無効か、または矛盾しています。

## SET\_INDEX\_STATS プロシージャ

このプロシージャは、索引に関連する情報を設定します。

### 構文

```
DBMS_STATS.SET_INDEX_STATS (  
    ownname  VARCHAR2,  
    indname  VARCHAR2,  
    partname VARCHAR2 DEFAULT NULL,  
    stattab  VARCHAR2 DEFAULT NULL,  
    statid   VARCHAR2 DEFAULT NULL,  
    numrows  NUMBER    DEFAULT NULL,  
    numlblks NUMBER    DEFAULT NULL,  
    numdist  NUMBER    DEFAULT NULL,  
    avglblk  NUMBER    DEFAULT NULL,  
    avgdblk  NUMBER    DEFAULT NULL,  
    clstfct  NUMBER    DEFAULT NULL,  
    indlevel NUMBER    DEFAULT NULL,  
    flags    NUMBER    DEFAULT NULL,  
    statown  VARCHAR2  DEFAULT NULL);
```

### パラメータ

表 49-5 SET\_INDEX\_STATS プロシージャのパラメータ

パラメータ	説明
ownname	スキーマ名。
indname	索引名。
partname	統計情報を格納する索引パーティションの名前。索引がパーティション化されていて、partname が NULL の場合、統計情報はグローバルな索引レベルで格納されます。
stattab	統計情報の格納場所を示すユーザー統計表の識別子。stattab が NULL の場合、統計情報はディクショナリ内に直接格納されます。
statid	stattab 内の統計情報を関連付ける識別子（オプション）（stattab が NULL でない場合のみ該当します）。
numrows	索引（パーティション）内の行数。
numlblks	索引（パーティション）内のリーフ・ブロックの数。
numdist	索引（パーティション）内の個別キーの数。

表 49-5 SET\_INDEX\_STATS プロシージャのパラメータ

パラメータ	説明
avglblk	この索引（パーティション）について各個別キーが出現するリーフ・ブロックの平均整数値。この値が提供されない場合、この値は numlblks と numdist から導出されます。
avgdblk	この索引（パーティション）について個別キーが指す表内のデータ・ブロックの平均整数値。この値が提供されない場合、この値は clstfct と numdist から導出されます。
clstfct	user_indexes ビューの clustering_factor 列の説明を参照してください。
indlevel	索引（パーティション）の高さ。
flags	Oracle 内部で使用（NULL のままにします）。
statown	stattab を含んだスキーマ（ownname と異なる場合）。

例外

ORA-20000: オブジェクトが存在しないか、または権限が不十分です。

ORA-20001: 入力値が無効です。

SET\_TABLE\_STATS プロシージャ

このプロシージャは、表に関連する情報を設定します。

構文

```
DBMS_STATS.SET_TABLE_STATS (  
  ownname  VARCHAR2,  
  tabname  VARCHAR2,  
  partname VARCHAR2 DEFAULT NULL,  
  stattab  VARCHAR2 DEFAULT NULL,  
  statid   VARCHAR2 DEFAULT NULL,  
  numrows  NUMBER   DEFAULT NULL,  
  numblks  NUMBER   DEFAULT NULL,  
  avgrlen  NUMBER   DEFAULT NULL,  
  flags    NUMBER   DEFAULT NULL,  
  statown  VARCHAR2 DEFAULT NULL);
```



パラメータ

表 49-6 SET\_TABLE\_STATS プロシージャのパラメータ

パラメータ	説明
ownname	スキーマ名。
tabname	表名。
partname	統計情報を格納する表パーティションの名前。表がパーティション化されていて、partname が NULL の場合、統計情報はグローバルな表レベルで格納されます。
stattab	統計情報の格納場所を示すユーザー統計表の識別子。stattab が NULL の場合、統計情報はディクショナリ内に直接格納されます。
statid	stattab 内の統計情報を関連付ける識別子（オプション）（stattab が NULL でない場合のみ該当します）。
numrows	表（パーティション）内の行数。
numblks	表（パーティション）が占有するブロックの数。
avgrlen	表（パーティション）の行の平均長。
flags	Oracle 内部で使用（NULL のままにします）。
statown	stattab を含んだスキーマ（ownname と異なる場合）。

例外

ORA-20000: オブジェクトが存在しないか、または権限が不十分です。

ORA-20001: 入力値が無効です。

CONVERT\_RAW\_VALUE プロシージャ

このプロシージャは、最大値または最小値の内部表記をデータ型固有値に変換します。GET\_COLUMN\_STATS または PREPARE\_COLUMN\_VALUES で入力された StatRec 構造の minval フィールドと maxval フィールドの値が、有効な入力値です。

構文

```
DBMS_STATS.CONVERT_RAW_VALUE (
    rawval      RAW,
    resval OUT VARCHAR2);

DBMS_STATS.CONVERT_RAW_VALUE (
    rawval      RAW,
    resval OUT DATE);

DBMS_STATS.CONVERT_RAW_VALUE (
    rawval      RAW,
    resval OUT NUMBER);

DBMS_STATS.CONVERT_RAW_VALUE_NVARCHAR (
    rawval      RAW,
    resval OUT NVARCHAR2);

DBMS_STATS.CONVERT_RAW_VALUE_ROWID (
    rawval      RAW,
    resval OUT ROWID);
```

プラグマ

```
pragma restrict_references(convert_raw_value, WNDS, RNDS, WNPS, RNPS);
pragma restrict_references(convert_raw_value_nvarchar, WNDS, RNDS, WNPS, RNPS);
pragma restrict_references(convert_raw_value_rowid, WNDS, RNDS, WNPS, RNPS);
```

パラメータ

表 49-7 CONVERT\_RAW\_VALUE プロシージャのパラメータ

パラメータ	説明
rawval	列最大または列最小のデータ型固有アウトプット・パラメータの内部表記。
resval	変換済みの型固有値。

例外

なし。

# GET\_COLUMN\_STATS プロシージャ

このプロシージャは、列に関連するすべての情報を取得します。

## 構文

```
DBMS_STATS.GET_COLUMN_STATS (  
    ownname      VARCHAR2,  
    tabname      VARCHAR2,  
    colname      VARCHAR2,  
    partname     VARCHAR2 DEFAULT NULL,  
    stattab      VARCHAR2 DEFAULT NULL,  
    statid       VARCHAR2 DEFAULT NULL,  
    distcnt OUT NUMBER,  
    density OUT NUMBER,  
    nullcnt OUT NUMBER,  
    srec      OUT StatRec,  
    avgclen OUT NUMBER,  
    statown   VARCHAR2 DEFAULT NULL);
```

## パラメータ

表 49-8 GET\_COLUMN\_STATS プロシージャのパラメータ

パラメータ	説明
ownname	スキーマ名。
tabname	この列が所属する表の名前。
colname	列名。
partname	統計情報を取得する表パーティションの名前。表がパーティション化されていて、partname が NULL の場合、統計情報はグローバルな表レベルで取り出されます。
stattab	統計情報を取り出す場所を示すユーザー統計表の識別子。stattab が NULL の場合、統計情報はディクショナリから直接取り出されます。
statid	stattab 内の統計情報を関連付ける識別子（オプション）（stattab が NULL でない場合のみ該当します）。
distcnt	個別値の数。
density	列密度。
nullcnt	NULL の数。
srec	列最大、列最小およびヒストグラム値の内部表記を保持する構造。

表 49-8 GET\_COLUMN\_STATS プロシージャのパラメータ

パラメータ	説明
avgclen	列の平均長（単位はバイト）。
statown	stattab を含んだスキーマ（ownname と異なる場合）。

例外

ORA-20000: オブジェクトが存在しないか権限が不十分、または要求されたオブジェクトの統計情報が格納されていません。

GET\_INDEX\_STATS プロシージャ

このプロシージャは、索引に関連するすべての情報を取得します。

構文

```
DBMS_STATS.GET_INDEX_STATS (  
    ownname      VARCHAR2,  
    indname      VARCHAR2,  
    partname     VARCHAR2 DEFAULT NULL,  
    stattab      VARCHAR2 DEFAULT NULL,  
    statid       VARCHAR2 DEFAULT NULL,  
    numrows     OUT  NUMBER,  
    numlblks    OUT  NUMBER,  
    numdist     OUT  NUMBER,  
    avglblk     OUT  NUMBER,  
    avgdblk     OUT  NUMBER,  
    clstfct     OUT  NUMBER,  
    indlevel    OUT  NUMBER,  
    statown     VARCHAR2 DEFAULT NULL);
```

パラメータ

表 49-9 GET\_INDEX\_STATS プロシージャのパラメータ

パラメータ	説明
ownname	スキーマ名。
indname	索引名。
partname	統計情報を取得する索引パーティションの名前。索引がパーティション化されていて、partname が NULL の場合、統計情報はグローバルな索引レベルで取り出されます。

表 49-9 GET\_INDEX\_STATS プロシージャのパラメータ

パラメータ	説明
stattab	統計情報を取り出す場所を示すユーザー統計表の識別子。 stattab が NULL の場合、統計情報はディクショナリから直接取り出されます。
statid	stattab 内の統計情報を関連付ける識別子 (オプション) (stattab が NULL でない場合のみ該当します)。
numrows	索引 (パーティション) 内の行数。
numlblks	索引 (パーティション) 内のリーフ・ブロックの数。
numdist	索引 (パーティション) 内の個別キーの数。
avglblk	この索引 (パーティション) について各個別キーが出現するリーフ・ブロックの平均整数値。
avgdblk	この索引 (パーティション) について個別キーが指す表内のデータ・ブロックの平均整数値。
clstfct	索引 (パーティション) のクラスタ化要素。
indlevel	索引 (パーティション) の高さ。
statown	stattab を含んだスキーマ (ownname と異なる場合)。

例外

ORA-20000: オブジェクトが存在しないか権限が不十分、または要求したオブジェクトの統計情報が格納されていません。

GET\_TABLE\_STATS プロシージャ

このプロシージャは、表に関連するすべての情報を取得します。

構文

```
DBMS_STATS.GET_TABLE_STATS (  
  ownname      VARCHAR2,  
  tabname      VARCHAR2,  
  partname     VARCHAR2 DEFAULT NULL,  
  stattab      VARCHAR2 DEFAULT NULL,  
  statid       VARCHAR2 DEFAULT NULL,  
  numrows OUT  NUMBER,  
  numlblks OUT  NUMBER,  
  avgrlen OUT  NUMBER,  
  statown     VARCHAR2 DEFAULT NULL);
```

パラメータ

表 49-10 GET\_TABLE\_STATS プロシージャのパラメータ

パラメータ	説明
ownname	スキーマ名。
tabname	列が所属する表の名前。
partname	統計情報を取得する表パーティションの名前。表がパーティション化されていて、partname が NULL の場合、統計情報はグローバルな表レベルで取り出されます。
stattab	統計情報を取り出す場所を示すユーザー統計表の識別子。 stattab が NULL の場合、統計情報はディクショナリから直接取り出されます。
statid	stattab 内の統計情報を関連付ける識別子（オプション） （stattab が NULL でない場合のみ該当します）。
numrows	表（パーティション）内の行数。
numblks	表（パーティション）が占有するブロックの数。
avgrlen	表（パーティション）の行の平均長。
statown	stattab を含んだスキーマ（ownname と異なる場合）。

例外

ORA-20000: オブジェクトが存在しないか権限が不十分、または要求されたオブジェクトの統計情報が格納されていません。

DELETE\_COLUMN\_STATS プロシージャ

このプロシージャは、列に関連する統計情報を削除します。

構文

```
DBMS_STATS.DELETE_COLUMN_STATS (  
  ownname      VARCHAR2,  
  tabname      VARCHAR2,  
  colname      VARCHAR2,  
  partname     VARCHAR2 DEFAULT NULL,  
  stattab      VARCHAR2 DEFAULT NULL,  
  statid       VARCHAR2 DEFAULT NULL,  
  cascade_parts BOOLEAN  DEFAULT TRUE,  
  statown      VARCHAR2 DEFAULT NULL);
```

パラメータ

表 49-11 DELETE\_COLUMN\_STATS プロシージャのパラメータ

パラメータ	説明
ownname	スキーマ名。
tabname	列が所属する表の名前。
colname	列名。
partname	統計情報を削除する表パーティションの名前。表がパーティション化されていて、partname が NULL の場合、グローバルな列統計情報が削除されます。
stattab	統計情報を削除する場所を示すユーザー統計表の識別子。stattab が NULL の場合、統計情報はディクショナリから直接削除されます。
statid	stattab 内の統計情報を関連付ける識別子（オプション）（stattab が NULL でない場合のみ該当します）。
cascade_parts	表がパーティション化されていて、partname が NULL の場合、このパラメータを TRUE に設定すると、基礎となるすべてのパーティションについても列の統計情報が削除されます。
statown	stattab を含んだスキーマ（ownname と異なる場合）。

例外

ORA-20000: オブジェクトが存在しないか、または権限が不十分です。

DELETE\_INDEX\_STATS プロシージャ

このプロシージャは、索引に関連する統計情報を削除します。

構文

```
DBMS_STATS.DELETE_INDEX_STATS (  
  ownname      VARCHAR2,  
  indname      VARCHAR2,  
  partname     VARCHAR2 DEFAULT NULL,  
  stattab      VARCHAR2 DEFAULT NULL,  
  statid       VARCHAR2 DEFAULT NULL,  
  cascade_parts BOOLEAN  DEFAULT TRUE,  
  statown      VARCHAR2 DEFAULT NULL);
```

パラメータ

表 49-12 DELETE\_INDEX\_STATS プロシージャのパラメータ

パラメータ	説明
ownname	スキーマ名。
indname	索引名。
partname	統計情報を削除する索引パーティションの名前。索引がパーティション化されていて、partname が NULL の場合、索引統計情報はグローバル・レベルで削除されます。
stattab	統計情報を削除する場所を示すユーザー統計表の識別子。stattab が NULL の場合、統計情報はディクショナリから直接削除されます。
statid	stattab 内の統計情報を関連付ける識別子（オプション）（stattab が NULL でない場合のみ該当します）。
cascade_parts	索引がパーティション化されていて、partname が NULL の場合、このパラメータを TRUE に設定すると、基礎となるすべてのパーティションについても索引の統計情報が削除されます。
statown	stattab を含んだスキーマ（ownname と異なる場合）。

例外

ORA-20000: オブジェクトが存在しないか、または権限が不十分です。

DELETE\_TABLE\_STATS プロシージャ

このプロシージャは、表に関連する統計情報を削除します。

構文

```
DBMS_STATS.DELETE_TABLE_STATS (  
  ownname          VARCHAR2,  
  tabname           VARCHAR2,  
  partname          VARCHAR2  DEFAULT NULL,  
  stattab           VARCHAR2  DEFAULT NULL,  
  statid            VARCHAR2  DEFAULT NULL,  
  cascade_parts     BOOLEAN    DEFAULT TRUE,  
  cascade_columns   BOOLEAN    DEFAULT TRUE,  
  cascade_indexes   BOOLEAN    DEFAULT TRUE,  
  statown           VARCHAR2  DEFAULT NULL);
```



パラメータ

表 49-13 DELETE\_TABLE\_STATS プロシージャのパラメータ

パラメータ	説明
ownname	スキーマ名。
tabname	列が所属する表の名前。
colname	列名。
partname	統計情報を削除する表パーティションの名前。表がパーティション化されていて、partname が NULL の場合、統計情報はグローバルな表レベルで削除されます。
stattab	統計情報を削除する場所を示すユーザー統計表の識別子。 stattab が NULL の場合、統計情報はディクショナリから直接削除されます。
statid	stattab 内の統計情報を関連付ける識別子 (オプション) (stattab が NULL でない場合のみ該当します)。
cascade_parts	表がパーティション化されていて、partname が NULL の場合、このパラメータを TRUE に設定すると、基礎となるすべてのパーティションについても表の統計情報が削除されます。
cascade_columns	基礎となるすべての列について、DELETE_COLUMN_STATS をコールする必要があることを示します (cascade_parts パラメータを渡します)。
cascade_indexes	基礎となるすべての索引について、DELETE_INDEX_STATS をコールする必要があることを示します (cascade_parts パラメータを渡します)。
statown	stattab を含んだスキーマ (ownname と異なる場合)。

例外

ORA-20000: オブジェクトが存在しないか、または権限が不十分です。

## DELETE\_SCHEMA\_STATS プロシージャ

このプロシージャは、スキーマ全体の統計情報を削除します。

### 構文

```
DBMS_STATS.DELETE_SCHEMA_STATS (  
    ownname VARCHAR2,  
    stattab VARCHAR2 DEFAULT NULL,  
    statid VARCHAR2 DEFAULT NULL,  
    statown VARCHAR2 DEFAULT NULL);
```

### パラメータ

表 49-14 DELETE\_SCHEMA\_STATS プロシージャのパラメータ

パラメータ	説明
ownname	スキーマ名。
stattab	統計情報を削除する場所を示すユーザー統計表の識別子。 stattab が NULL の場合、統計情報はディクショナリから直接削除されます。
statid	stattab 内の統計情報を関連付ける識別子（オプション） （stattab が NULL でない場合のみ該当します）。
statown	stattab を含んだスキーマ（ownname と異なる場合）。

### 例外

ORA-20000: オブジェクトが存在しないか、または権限が不十分です。

## DELETE\_DATABASE\_STATS プロシージャ

このプロシージャは、データベース全体の統計情報を削除します。

### 構文

```
DBMS_STATS.DELETE_DATABASE_STATS (  
    stattab VARCHAR2 DEFAULT NULL,  
    statid VARCHAR2 DEFAULT NULL,  
    statown VARCHAR2 DEFAULT NULL);
```

パラメータ

表 49-15 DELETE\_DATABASE\_STATS プロシージャのパラメータ

パラメータ	説明
stattab	統計情報を削除する場所を示すユーザー統計表の識別子。 stattab が NULL の場合、統計情報はディクショナリから直接削除されます。
statid	stattab 内の統計情報を関連付ける識別子 (オプション) (stattab が NULL でない場合のみ該当します)。
statown	stattab を含むスキーマ。stattab が NULL ではなく、 statown が NULL の場合は、データベース内のすべてのスキーマ に、同じ名前の stattab を持つユーザー統計表が含まれている とみなされます。

例外

ORA-20000: オブジェクトが存在しないか、または権限が不十分です。

統計情報の転送

次のプロシージャを使用して、ディクショナリからユーザー統計表へ (export\_\*) および  
ユーザー統計表からディクショナリへ (import\_\*) 統計情報を転送できます。

```
CREATE_STAT_TABLE
DROP_STAT_TABLE

EXPORT_COLUMN_STATS
EXPORT_INDEX_STATS
EXPORT_TABLE_STATS
EXPORT_SCHEMA_STATS
EXPORT_DATABASE_STATS

IMPORT_COLUMN_STATS
IMPORT_INDEX_STATS
IMPORT_TABLE_STATS
IMPORT_SCHEMA_STATS
IMPORT_DATABASE_STATS
```

## CREATE\_STAT\_TABLE プロシージャ

このプロシージャは、統計情報を保持できる ownname のスキーマにある stattab の名前で表を作成します。この表は、このパッケージのプロシージャを介して単独にアクセスされるため、この表を構成する列と型は互いに関係がありません。

### 構文

```
DBMS_STATS.CREATE_STAT_TABLE (
    ownname  VARCHAR2,
    stattab  VARCHAR2,
    tblspace VARCHAR2 DEFAULT NULL);
```

### パラメータ

表 49-16 CREATE\_STAT\_TABLE プロシージャのパラメータ

パラメータ	説明
ownname	スキーマ名。
stattab	作成する表の名前。ユーザーがディクショナリ統計情報を直接変更しない場合、この値は、stattab パラメータとして他のプロシージャに渡されます。
tblspace	統計表を作成する表領域。このパラメータを指定しないと、統計表はユーザーのデフォルトの表領域に作成されます。

### 例外

- ORA-20000: 表がすでに存在するか、権限が不十分です。
- ORA-20001: 表領域が存在しません。

## DROP\_STAT\_TABLE プロシージャ

このプロシージャは、ユーザー統計表を削除します。

### 構文

```
DBMS_STATS.DROP_STAT_TABLE (
    ownname VARCHAR2,
    stattab  VARCHAR2);
```

## パラメータ

表 49-17 DROP\_STAT\_TABLE プロシージャのパラメータ

パラメータ	説明
ownname	スキーマ名。
stattab	ユーザー統計表の識別子。

## 例外

ORA-20000: 表が存在しないか、または権限が不十分です。

## EXPORT\_COLUMN\_STATS プロシージャ

このプロシージャは、特定の列に関する統計情報を取り出し、stattab で識別されるユーザー統計表に格納します。

## 構文

```
DBMS_STATS.EXPORT_COLUMN_STATS (  
    ownname    VARCHAR2,  
    tabname    VARCHAR2,  
    colname    VARCHAR2,  
    partname   VARCHAR2 DEFAULT NULL,  
    stattab    VARCHAR2,  
    statid     VARCHAR2 DEFAULT NULL,  
    statown    VARCHAR2 DEFAULT NULL);
```

## パラメータ

表 49-18 EXPORT\_COLUMN\_STATS プロシージャのパラメータ

パラメータ	説明
ownname	スキーマ名。
tabname	列が所属する表の名前。
colname	列名。
partname	表パーティション名。表がパーティション化されていて、partname が NULL の場合、グローバルでパーティション化された列の統計情報がエクスポートされます。
stattab	統計情報の格納場所を示すユーザー統計表の識別子。
statid	stattab 内の統計情報を関連付ける識別子（オプション）。

表 49-18 EXPORT\_COLUMN\_STATS プロシージャのパラメータ

パラメータ	説明
statown	stattab を含んだスキーマ ( ownname と異なる場合 )。

例外

ORA-20000: オブジェクトが存在しないか、または権限が不十分です。

EXPORT\_INDEX\_STATS プロシージャ

このプロシージャは、特定の索引に関する統計情報を取り出し、stattab で識別されるユーザー統計表に格納します。

構文

```
DBMS_STATS.EXPORT_INDEX_STATS (
    ownname  VARCHAR2,
    indname  VARCHAR2,
    partname VARCHAR2 DEFAULT NULL,
    stattab  VARCHAR2,
    statid   VARCHAR2 DEFAULT NULL,
    statown  VARCHAR2 DEFAULT NULL);
```

パラメータ

表 49-19 EXPORT\_INDEX\_STATS プロシージャのパラメータ

パラメータ	説明
ownname	スキーマ名。
indname	索引名。
partname	索引パーティション名。索引がパーティション化されていて、partname が NULL の場合、グローバルでパーティション化された索引統計情報がエクスポートされます。
stattab	統計情報の格納場所を示すユーザー統計表の識別子。
statid	stattab 内の統計情報を関連付ける識別子 ( オプション )。
statown	stattab を含んだスキーマ ( ownname と異なる場合 )。

例外

ORA-20000: オブジェクトが存在しないか、または権限が不十分です。

## EXPORT\_TABLE\_STATS プロシージャ

このプロシージャは、特定の表に関する統計情報を取り出し、ユーザー統計表に格納します。cascadeを使用すると、指定した表に関連付けられている索引および列統計情報もすべてエクスポートされます。

### 構文

```
DBMS_STATS.EXPORT_TABLE_STATS (  
    ownname  VARCHAR2,  
    tabname  VARCHAR2,  
    partname VARCHAR2 DEFAULT NULL,  
    stattab  VARCHAR2,  
    statid   VARCHAR2 DEFAULT NULL,  
    cascade  BOOLEAN  DEFAULT TRUE,  
    statown  VARCHAR2 DEFAULT NULL);
```

### パラメータ

表 49-20 EXPORT\_TABLE\_STATS プロシージャのパラメータ

パラメータ	説明
ownname	スキーマ名。
tabname	表名。
partname	表パーティション名。表がパーティション化されていて、partname が NULL の場合、グローバルでパーティション化された表統計情報がエクスポートされます。
stattab	統計情報の格納場所を示すユーザー統計表の識別子。
statid	stattab 内の統計情報を関連付ける識別子（オプション）。
cascade	TRUE の場合は、この表の列と索引の統計情報もまたエクスポートされます。
statown	stattab を含んだスキーマ（ownname と異なる場合）。

### 例外

ORA-20000: オブジェクトが存在しないか、または権限が不十分です。

## EXPORT\_SCHEMA\_STATS プロシージャ

このプロシージャは、ownname で識別されるスキーマ内のすべてのオブジェクトに関する統計情報を取り出し、stattab で識別されるユーザー統計表に格納します。

### 構文

```
DBMS_STATS.EXPORT_SCHEMA_STATS (  
    ownname VARCHAR2,  
    stattab VARCHAR2,  
    statid   VARCHAR2 DEFAULT NULL,  
    statown  VARCHAR2 DEFAULT NULL);
```

### パラメータ

表 49-21 EXPORT\_SCHEMA\_STATS プロシージャのパラメータ

パラメータ	説明
ownname	スキーマ名。
stattab	統計情報の格納場所を示すユーザー統計表の識別子。
statid	stattab 内の統計情報を関連付ける識別子（オプション）。
statown	stattab を含んだスキーマ（ownname と異なる場合）。

### 例外

ORA-20000: オブジェクトが存在しないか、または権限が不十分です。

## EXPORT\_DATABASE\_STATS プロシージャ

このプロシージャは、データベース内のすべてのオブジェクトに関する統計情報を取り出し、statown.stattab で識別されるユーザー統計表に格納します。

### 構文

```
DBMS_STATS.EXPORT_DATABASE_STATS (  
    stattab VARCHAR2,  
    statid   VARCHAR2 DEFAULT NULL,  
    statown  VARCHAR2 DEFAULT NULL);
```



## パラメータ

表 49-22 EXPORT\_DATABASE\_STATS プロシージャのパラメータ

パラメータ	説明
stattab	統計情報の格納場所を示すユーザー統計表の識別子。
statid	stattab 内の統計情報を関連付ける識別子 (オプション)。
statown	stattab を含むスキーマ。statown が NULL の場合、データベース内のすべてのスキーマには stattab の名前を持つユーザー統計表が含まれているとみなされます。

## 例外

ORA-20000: オブジェクトが存在しないか、または権限が不十分です。

## IMPORT\_COLUMN\_STATS プロシージャ

このプロシージャは、stattab で識別されるユーザー統計表から特定の列に関する統計情報を取り出し、ディクショナリに格納します。

## 構文

```
DBMS_STATS.IMPORT_COLUMN_STATS (  
    ownname  VARCHAR2,  
    tabname  VARCHAR2,  
    colname  VARCHAR2,  
    partname VARCHAR2 DEFAULT NULL,  
    stattab  VARCHAR2,  
    statid   VARCHAR2 DEFAULT NULL,  
    statown  VARCHAR2 DEFAULT NULL);
```

## パラメータ

表 49-23 IMPORT\_COLUMN\_STATS プロシージャのパラメータ

パラメータ	説明
ownname	スキーマ名。
tabname	列が所属する表の名前。
colname	列名。
partname	表パーティション名。表がパーティション化されていて、partname が NULL の場合、グローバルでパーティション化された列統計情報がインポートされます。

表 49-23 IMPORT\_COLUMN\_STATS プロシージャのパラメータ

パラメータ	説明
stattab	統計情報を取り出す場所を示すユーザー統計表の識別子。
statid	stattab 内の統計情報を関連付ける識別子（オプション）。
statown	stattab を含んだスキーマ（ownname と異なる場合）。

例外

ORA-20000: オブジェクトが存在しないか、または権限が不十分です。

ORA-20001: ユーザー統計表の値が無効または矛盾しています。

IMPORT\_INDEX\_STATS プロシージャ

このプロシージャは、stattab で識別されるユーザー統計表から特定の索引に関する統計情報を取り出し、ディクショナリに格納します。

構文

```
DBMS_STATS.IMPORT_INDEX_STATS (  
    ownname  VARCHAR2,  
    indname  VARCHAR2,  
    partname VARCHAR2 DEFAULT NULL,  
    stattab  VARCHAR2,  
    statid   VARCHAR2 DEFAULT NULL,  
    statown  VARCHAR2 DEFAULT NULL);
```

パラメータ

表 49-24 IMPORT\_INDEX\_STATS プロシージャのパラメータ

パラメータ	説明
ownname	スキーマ名。
indname	索引名。
partname	索引パーティション名。索引がパーティション化されていて、partname が NULL の場合、グローバルでパーティション化された索引統計情報がインポートされます。
stattab	統計情報を取り出す場所を示すユーザー統計表の識別子。
statid	stattab 内の統計情報を関連付ける識別子（オプション）。
statown	stattab を含んだスキーマ（ownname と異なる場合）。

例外

- ORA-20000: オブジェクトが存在しないか、または権限が不十分です。
- ORA-20001: ユーザー統計表の値が無効または矛盾しています。

IMPORT\_TABLE\_STATS プロシージャ

このプロシージャは、stattab で識別されるユーザー統計表から特定の表に関する統計情報を取り出し、ディクショナリに格納します。cascade を使用すると、指定した表に関連付けられている索引および列統計情報もすべてインポートされます。

構文

```
DBMS_STATS.IMPORT_TABLE_STATS (  
    ownname  VARCHAR2,  
    tabname  VARCHAR2,  
    partname VARCHAR2 DEFAULT NULL,  
    stattab  VARCHAR2,  
    statid   VARCHAR2 DEFAULT NULL,  
    cascade  BOOLEAN  DEFAULT TRUE,  
    statown  VARCHAR2 DEFAULT NULL);
```

パラメータ

表 49-25 IMPORT\_TABLE\_STATS プロシージャのパラメータ

パラメータ	説明
ownname	スキーマ名。
tabname	表名。
partname	表パーティション名。表がパーティション化されていて、partname が NULL の場合、グローバルでパーティション化された表統計情報がインポートされます。
stattab	統計情報を取り出す場所を示すユーザー統計表の識別子。
statid	stattab 内の統計情報を関連付ける識別子（オプション）。
cascade	TRUE の場合は、この表の列と索引の統計情報もまたインポートされます。
statown	stattab を含んだスキーマ（ownname と異なる場合）。

例外

- ORA-20000: オブジェクトが存在しないか、または権限が不十分です。

ORA-20001: ユーザー統計表の値が無効または矛盾しています。

IMPORT\_SCHEMA\_STATS プロシージャ

このプロシージャは、ownname で識別されるスキーマ内のすべてのオブジェクトに関する統計情報をユーザー統計表から取り出し、ディクショナリに格納します。

構文

```
DBMS_STATS.IMPORT_SCHEMA_STATS (  
    ownname VARCHAR2,  
    stattab VARCHAR2,  
    statid   VARCHAR2 DEFAULT NULL,  
    statown  VARCHAR2 DEFAULT NULL);
```

パラメータ

表 49-26 IMPORT\_SCHEMA\_STATS プロシージャのパラメータ

パラメータ	説明
ownname	スキーマ名。
stattab	統計情報を取り出す場所を示すユーザー統計表の識別子。
statid	stattab 内の統計情報を関連付ける識別子 (オプション)。
statown	stattab を含んだスキーマ (ownname と異なる場合)。

例外

ORA-20000: オブジェクトが存在しないか、または権限が不十分です。

ORA-20001: ユーザー統計表の値が無効または矛盾しています。

IMPORT\_DATABASE\_STATS プロシージャ

このプロシージャは、データベース内のすべてのオブジェクトに関する統計情報をユーザー統計表から取り出し、ディクショナリに格納します。

構文

```
DBMS_STATS.IMPORT_DATABASE_STATS (  
    stattab VARCHAR2,  
    statid   VARCHAR2 DEFAULT NULL,  
    statown  VARCHAR2 DEFAULT NULL);
```

## パラメータ

表 49-27 IMPORT\_DATABASE\_STATS プロシージャのパラメータ

パラメータ	説明
stattab	統計情報を取り出す場所を示すユーザー統計表の識別子。
statid	stattab 内の統計情報を関連付ける識別子（オプション）。
statown	stattab を含むスキーマ。statown が NULL の場合、データベース内のすべてのスキーマには stattab の名前を持つユーザー統計表が含まれているとみなされます。

## 例外

- ORA-20000: オブジェクトが存在しないか、または権限が不十分です。
- ORA-20001: ユーザー統計表の値が無効または矛盾しています。

## オプティマイザ統計情報の収集

次のプロシージャを使用すると、特定のクラスのオプティマイザ統計情報を収集でき、ANALYZE コマンドによってパフォーマンスの向上が可能になります。

```
GATHER_INDEX_STATS
GATHER_TABLE_STATS
GATHER_SCHEMA_STATS
GATHER_DATABASE_STATS
```

statown、stattab および statid パラメータは、パッケージに対して、新規の統計情報を収集する前に、指定した表内の現行の統計情報をバックアップするように指示します。

関連オブジェクトに十分な統計情報がある場合、導出オブジェクトの統計情報を生成するために、Oracle は次のプロシージャも提供します。

```
GENERATE_STATS
```

## GATHER\_INDEX\_STATS プロシージャ

このプロシージャは、索引の統計情報を収集します。これは、ANALYZE INDEX [ownname.]indname [PARTITION partname] COMPUTE STATISTICS | ESTIMATE STATISTICS SAMPLE estimate\_percent PERCENT を実行するのと同じです。

並列的には実行しません。

構文

```
DBMS_STATS_GATHER_INDEX_STATS (  
    ownname          VARCHAR2,  
    indname          VARCHAR2,  
    partname         VARCHAR2 DEFAULT NULL,  
    estimate_percent NUMBER  DEFAULT NULL,  
    stattab          VARCHAR2 DEFAULT NULL,  
    statid           VARCHAR2 DEFAULT NULL,  
    statown          VARCHAR2 DEFAULT NULL);
```

パラメータ

表 49-28 GATHER\_INDEX\_STATS プロシージャのパラメータ

パラメータ	説明
ownname	分析する索引のスキーマ。
indname	索引名。
partname	パーティション名。
estimate_percent	推定する行のパーセント（NULL は計算を意味します）。有効な範囲は、0.000001~100 です。この値は、よい結果をアーカイブするために自動的に増加できます。
stattab	現行の統計情報を保存する場所を示すユーザー統計表の識別子。
statid	stattab 内の統計情報を関連付ける識別子（オプション）。
statown	stattab を含んだスキーマ（ownname と異なる場合）。

例外

- ORA-20000: 索引が存在しないか、または権限が不十分です。
- ORA-20001: 入力値が無効です。

GATHER\_TABLE\_STATS プロシージャ

このプロシージャは、表と列（および索引）の統計情報を収集します。このプロシージャは、可能な限り多くの作業を並列化しますが、個々のパラメータで説明するように、いくつかの制限があります。分析する表に対する SELECT 権限がユーザーにない場合、この操作は並列化しません。

構文

```
DBMS_STATS.GATHER_TABLE_STATS (
    ownname          VARCHAR2,
    tabname           VARCHAR2,
    partname          VARCHAR2 DEFAULT NULL,
    estimate_percent  NUMBER    DEFAULT NULL,
    block_sample      BOOLEAN    DEFAULT FALSE,
    method_opt        VARCHAR2  DEFAULT 'FOR ALL COLUMNS SIZE 1',
    degree            NUMBER    DEFAULT NULL,
    granularity       VARCHAR2  DEFAULT 'DEFAULT',
    cascade           BOOLEAN    DEFAULT FALSE,
    stattab           VARCHAR2  DEFAULT NULL,
    statid            VARCHAR2  DEFAULT NULL,
    statown           VARCHAR2  DEFAULT NULL);
```

パラメータ

表 49-29 GATHER\_TABLE\_STATS プロシージャのパラメータ

パラメータ	説明
ownname	分析する表のスキーマ。
tabname	表名。
partname	パーティション名。
estimate_percent	推定する行のパーセント (NULL は計算を意味します)。有効な範囲は、0.000001~100 です。この値は、よい結果をアーカイブするために自動的に増加できます。
block_sample	ランダム行サンプリングのかわりにランダム・ブロック・サンプリングを使用するかどうかを示します。ランダム・ブロック・サンプリングがより効率的ですが、データがディスク上にランダムに分散していない場合、サンプル値はある程度相関があります。統計情報の推定を行った場合のみ該当します。
method_opt	次の書式の方法オプション (句 'SIZE 1' は、並列的に統計情報を収集し非表示の句を使用するために必要です)。 FOR ALL [INDEXED   HIDDEN] COLUMNS [SIZE integer] FOR COLUMNS [SIZE integer] column attribute [,column attribute ...] オブティマイザに関連する表統計情報は、常に収集されます。
degree	並列度 (NULL は、表のデフォルト値の使用を意味します)。

表 49-29 GATHER\_TABLE\_STATS プロシージャのパラメータ

パラメータ	説明
granularity	収集する統計情報の細分化（表がパーティション化されている場合のみ該当します）。  DEFAULT: グローバル・レベルでパーティション・レベルの統計情報を収集します。  SUBPARTITION: サブパーティション・レベルの統計情報を収集します。  PARTITION: パーティション・レベルの統計情報を収集します。  GLOBAL: グローバルな統計情報を収集します。  ALL: すべての統計情報（サブパーティション、パーティションおよびグローバル）を収集します。
cascade	表の索引について統計情報を収集します。索引統計情報の収集は並列化されていません。このオプションを使用することは、表の各索引で gather_index_stats プロシージャを実行するのと同じです。
stattab	現行の統計情報を保存する場所を示すユーザー統計表の識別子。
statid	stattab 内の統計情報を関連付ける識別子（オプション）。
statown	stattab を含んだスキーマ（ownname と異なる場合）。

例外

ORA-20000: 表が存在しないか、または権限が不十分です。

ORA-20001: 入力値が無効です。

GATHER\_SCHEMA\_STATS プロシージャ

このプロシージャは、スキーマ内のすべてのオブジェクトに関する統計情報を収集します。

構文

```
DBMS_STATS.GATHER_SCHEMA_STATS (  
  ownname          VARCHAR2,  
  estimate_percent NUMBER  DEFAULT NULL,  
  block_sample     BOOLEAN  DEFAULT FALSE,  
  method_opt       VARCHAR2 DEFAULT 'FOR ALL COLUMNS SIZE 1',  
  degree           NUMBER   DEFAULT NULL,  
  granularity      VARCHAR2 DEFAULT 'DEFAULT',  
  cascade          BOOLEAN  DEFAULT FALSE);
```



```

DBMS_STATS.GATHER_SCHEMA_STATS (
    ownname          VARCHAR2,
    estimate_percent NUMBER  DEFAULT NULL,
    block_sample     BOOLEAN  DEFAULT FALSE,
    method_opt       VARCHAR2 DEFAULT 'FOR ALL COLUMNS SIZE 1',
    degree           NUMBER   DEFAULT NULL,
    granularity      VARCHAR2 DEFAULT 'DEFAULT',
    cascade          BOOLEAN  DEFAULT FALSE,
    stattab          VARCHAR2 DEFAULT NULL,
    statid           VARCHAR2 DEFAULT NULL,
    options          VARCHAR2 DEFAULT 'GATHER',
    objlist          OUT      ObjectTab,
    statown          VARCHAR2 DEFAULT NULL);

```

## パラメータ

**表 49-30 GATHER\_SCHEMA\_STATS プロシージャのパラメータ**

パラメータ	説明
ownname	分析するスキーマ ( NULL は現行スキーマを意味します )。
estimate_percent	推定する行のパーセント ( NULL は計算を意味します )。有効な範囲は、0.000001~100 です。
block_sample	ランダム行サンプリングのかわりにランダム・ブロック・サンプリングを使用するかどうかを示します。ランダム・ブロック・サンプリングがより効率的ですが、データがディスク上にランダムに分散していない場合、サンプル値はある程度相関があります。統計情報の推定を行った場合のみ該当します。
method_opt	次の書式の方法オプション ( 句 'SIZE 1' は、並列的に統計情報を収集し非表示の句を使用するために必要です )。 FOR ALL [INDEXED   HIDDEN] COLUMNS [SIZE integer] この値は、すべての個別表に渡されます。
degree	並列度 ( NULL は、表のデフォルト値の使用を意味します )。

表 49-30 GATHER\_SCHEMA\_STATS プロシージャのパラメータ

パラメータ	説明
granularity	<p>収集する統計情報の細分化（表がパーティション化されている場合のみ該当します）。</p> <p>DEFAULT: グローバル・レベルでパーティション・レベルの統計情報を収集します。</p> <p>SUBPARTITION: サブパーティション・レベルの統計情報を収集します。</p> <p>PARTITION: パーティション・レベルの統計情報を収集します。</p> <p>GLOBAL: グローバルな統計情報を収集します。</p> <p>ALL: すべての統計情報（サブパーティション、パーティションおよびグローバル）を収集します。</p>
cascade	<p>索引についても統計情報を収集します。</p> <p>索引統計情報の収集は並列化されません。このオプションを使用することは、表と列の統計情報の収集に加えて、スキーマ内の各索引で gather_index_stats プロシージャを実行するのと同じです。</p>
stattab	現行の統計情報を保存する場所を示すユーザー統計表の識別子。
statid	stattab 内の統計情報を関連付ける識別子（オプション）。
options	<p>統計情報を収集するオブジェクトの詳細は、次のように指定します。</p> <p>GATHER: スキーマ内のすべてのオブジェクトに関する統計情報を収集します。</p> <p>GATHER STALE: *_tab_modifications ビューを調べて判別した失効オブジェクトに関する統計情報を収集します。また、失効と判別されたオブジェクトのリストも戻します。</p> <p>GATHER EMPTY: 現在統計情報がないオブジェクトについて統計情報を収集し、統計情報なしと判別されたオブジェクトのリストも戻します。</p> <p>LIST STALE: *_tab_modifications ビューを調べて判別した失効オブジェクトのリストを戻します。</p> <p>LIST EMPTY: 現在統計情報がないオブジェクトのリストを戻します。</p>
objlist	失効または空と判別されたオブジェクトのリスト。
statown	stattab を含んだスキーマ（ownname と異なる場合）。

例外

ORA-20000: スキーマが存在しないか、または権限が不十分です。  
ORA-20001: 入力値が無効です。

GATHER\_DATABASE\_STATS プロシージャ

このプロシージャは、データベース内のすべてのオブジェクトに関する統計情報を収集します。

構文

```
DBMS_STATS.GATHER_DATABASE_STATS (  
    estimate_percent NUMBER    DEFAULT NULL,  
    block_sample      BOOLEAN  DEFAULT FALSE,  
    method_opt        VARCHAR2 DEFAULT 'FOR ALL COLUMNS SIZE 1',  
    degree            NUMBER    DEFAULT NULL,  
    granularity        VARCHAR2 DEFAULT 'DEFAULT',  
    cascade            BOOLEAN  DEFAULT FALSE);  
  
DBMS_STATS.GATHER_DATABASE_STATS (  
    estimate_percent NUMBER    DEFAULT NULL,  
    block_sample      BOOLEAN  DEFAULT FALSE,  
    method_opt        VARCHAR2 DEFAULT 'FOR ALL COLUMNS SIZE 1',  
    degree            NUMBER    DEFAULT NULL,  
    granularity        VARCHAR2 DEFAULT 'DEFAULT',  
    cascade            BOOLEAN  DEFAULT FALSE,  
    stattab           VARCHAR2 DEFAULT NULL,  
    statid            VARCHAR2 DEFAULT NULL,  
    options           VARCHAR2 DEFAULT 'GATHER',  
    objlist           OUT      ObjectTab,  
    statown           VARCHAR2 DEFAULT NULL);
```

パラメータ

表 49-31 GATHER\_DATABASE\_STATS プロシージャのパラメータ

パラメータ	説明
estimate_percent	推定する行のパーセント（NULL は計算を意味します）。有効な範囲は、0.000001~100 です。
block_sample	ランダム行サンプリングのかわりにランダム・ブロック・サンプリングを使用するかどうかを示します。ランダム・ブロック・サンプリングがより効率的ですが、データがディスク上にランダムに分散していない場合、サンプル値はある程度相関があります。統計情報の推定を行った場合のみ該当します。

表 49-31 GATHER\_DATABASE\_STATS プロシージャのパラメータ

パラメータ	説明
method_opt	次の書式の方法オプション（句 'SIZE 1' は、並列的に統計情報を収集し非表示の句を使用するために必要です）。 FOR ALL [INDEXED   HIDDEN] COLUMNS [SIZE integer] この値は、すべての個別表に渡されます。
degree	並列度（NULL は、表のデフォルト値の使用を意味します）。
granularity	収集する統計情報の細分化（表がパーティション化されている場合のみ該当します）。 DEFAULT: グローバル・レベルでパーティション・レベルの統計情報を収集します。 SUBPARTITION: サブパーティション・レベルの統計情報を収集します。 PARTITION: パーティション・レベルの統計情報を収集します。 GLOBAL: グローバルな統計情報を収集します。 ALL: すべての統計情報（サブパーティション、パーティションおよびグローバル）を収集します。
cascade	索引についても統計情報を収集します。索引統計情報の収集は並列化されません。このオプションを使用することは、表と列の統計情報の収集に加えて、データベース内の各索引で gather_index_stats プロシージャを実行するのと同じです。
stattab	現行の統計情報を保存する場所を示すユーザー統計表の識別子。 統計表は、分析するオブジェクト同じスキーマに常駐するとみなされるので、各スキーマにこのオプションを使用するための表が 1 つ必要です。
statid	stattab 内の統計情報を関連付ける識別子（オプション）。
options	統計情報を収集するオブジェクトの詳細は、次のように指定します。 GATHER STALE: *_tab_modifications ビューを調べて判別した失効オブジェクトについて、統計情報を収集します。また、失効と判別されたオブジェクトのリストも戻します。 GATHER EMPTY: 現在統計情報がないオブジェクトについて統計情報を収集し、統計情報なしと判別されたオブジェクトのリストも戻します。 LIST STALE: *_tab_modifications ビューを調べて判別した失効オブジェクトのリストを戻します。 LIST EMPTY: 現在統計情報がないオブジェクトのリストを戻します。

表 49-31 GATHER\_DATABASE\_STATS プロシージャのパラメータ

パラメータ	説明
objlist	失効または空と判別されたオブジェクトのリスト。
statown	stattab を含んだスキーマ ( ownname と異なる場合 )。

例外

ORA-20000: 権限が不十分です。

ORA-20001: 入力値が無効です。

GENERATE\_STATS プロシージャ

このプロシージャは、関連するオブジェクトで以前に収集した統計情報から、オブジェクトの統計情報を生成します。完全に移入されたスキーマについては、より正確な統計情報が必要な場合は、GATHER プロシージャをかわりに使用する必要があります。現在サポートされているオブジェクトは、B ツリー索引とビットマップ索引です。

構文

```
DBMS_STATS.GENERATE_STATS (
    ownname    VARCHAR2,
    objname    VARCHAR2,
    organized  NUMBER DEFAULT 7);
```

パラメータ

表 49-32 GENERATE\_STATS プロシージャのパラメータ

パラメータ	説明
ownname	オブジェクトのスキーマ。
objname	オブジェクト名。
organized	索引とその基礎となる表の間で関連付けられた順位付けの量。複雑に編成されている索引には、ディスク上の連続行を参照する連続索引キーがその表 ( 同じブロック ) に対してあります。複雑に編成されていない索引には、ディスク上の異なる表ブロックを参照する連続キーがあります。  このパラメータは、B ツリー索引に対してのみ使用します。数値は、0 ~ 10 の範囲で使用でき、0 は完全に編成された索引、10 は完全に編成解除された索引を示します。

## 例外

ORA-20000: サポートされていないオブジェクト型で、オブジェクトは存在しません。

ORA-20001: 無効なオプションまたは無効な統計情報です。

## 例

**使用例** 前回統計情報を収集してから、emp 表に対して大量の変更がありました。コストベースのオプティマイザが最適プランを選択するために、統計情報を再度収集する必要があります。しかし、ユーザーは、現行プランが許容されると、新規の統計情報によってオプティマイザが誤ったプランを選択することを懸念しています。ユーザーは、次のように指定できます。

```
BEGIN
  DBMS_STATS.CREATE_STAT_TABLE ('scott', 'savestats');
  DBMS_STATS.GATHER_TABLE_STATS ('scott', 'emp', 5, stattab => 'savestats');
END;
```

この操作は、新規統計情報を emp 表に収集しますが、最初に、元の統計情報をユーザー統計表 emp.savestats に保存します。

ユーザーが、新規統計情報によってオプティマイザが誤ったプランを生成すると考えている場合は、元の統計情報を次のように復元できます。

```
BEGIN
  DBMS_STATS.DELETE_TABLE_STATS ('scott', 'emp');
  DBMS_STATS.IMPORT_TABLE_STATS ('scott', 'emp', stattab => 'savestats');
END;
```

---

## DBMS\_TRACE

Oracle8i PL/SQL は、サーバー上での PL/SQL プログラムの実行をトレースするために、API を提供します。サーバー上に DBMS\_TRACE パッケージとしてインプリメントされたトレース API を使用すると、PL/SQL ファンクション、プロシージャおよび例外をトレースできます。

DBMS\_TRACE は、セッションで PL/SQL トレースを開始および停止するサブプログラムを提供します。トレース・データはプログラムの実行時に収集され、Oracle Server のトレース・ファイルに書き出されます。

一般的なセッションには、次の処理が含まれます。

- セッションで PL/SQL トレースを開始します ( DBMS\_TRACE.SET\_PLSQL\_TRACE )。
- トレースするアプリケーションを実行します。
- セッションでの PL/SQL トレースを停止します ( DBMS\_TRACE.CLEAR\_PLSQL\_TRACE )。

---

## 要件

このパッケージは、SYS の下で作成する必要があります。

## 制限事項

マルチスレッド・サーバー (MTS) では、PL/SQL トレースを使用できません。

## 定数

```
trace_all_calls          constant INTEGER := 1;
trace_enabled_calls      constant INTEGER := 2;
trace_all_exceptions     constant INTEGER := 4;
trace_enabled_exceptions constant INTEGER := 8;
/*
 * The version of the trace package. These constants will change as the
 * package evolves. Use the PLSQL_TRACE_VERSION procedure (described below)
 * to get the current version of the package.
 */
trace_major_version      constant BINARY_INTEGER := 1;
trace_minor_version      constant BINARY_INTEGER := 0;
```

## DBMS\_TRACE の使用方法

### データ量の制御

大規模なアプリケーションをプロファイルすると、データ量が膨大になる可能性があります。トレース・データの収集に関する特定のプログラム・ユニットを使用可能にして、収集するデータ量を制御できます。

プログラム・ユニットは、コンパイルとデバッグを行って使用可能にできます。次のいずれかの方法で行います。

```
alter session set plsql_debug=true;
create or replace ... /* create the library units - debug information will be
generated */
```

または

```
/* recompile specific library unit with debug option */
alter [PROCEDURE | FUNCTION | PACKAGE BODY] <libunit-name> compile debug;
```



---

---

**注意：** 2 番目の方法は、無名ブロックに対しては使用できません。

---

---

## トレース・データの収集

**コールのトレース** 使用可能なコールのトレースには、次の 2 つのレベルがあります。

- レベル 1: すべてのコールをトレースします。これは、定数 `trace_all_calls` に対応します。
- レベル 2: 使用可能なプログラム・ユニットのみ、コールをトレースします。これは、定数 `trace_enabled_calls` に対応します。

リモート・プロシージャ・コール (RPC) については、使用可能かどうかを検出できないため、RPC ではレベル 1 でのみトレースできます。

**例外のトレース** 例外のトレースには、次の 2 つのレベルがあります。

- レベル 1: すべての例外をトレースします。これは、定数 `trace_all_exceptions` に対応します。
- レベル 2: 使用可能なプログラム・ユニットのみ、発生した例外をトレースします。これは、定数 `trace_enabled_exceptions` に対応します。

---

---

**注意：** コールと例外の両方のトレースとも、レベル 1 はレベル 2 を上書きします。たとえば、レベル 1 とレベル 2 の両方が使用可能な場合は、レベル 1 が優先します。

---

---

## 収集されたデータ

使用可能なプログラム・ユニットについてのみトレースが要求されていて、現行のプログラム・ユニットが使用可能ではない場合、トレース・データは書き込まれません。

現行のプログラム・ユニットが使用可能な場合、コールのトレースでは、プログラム・ユニットの型、名前およびスタックの深さが書き込まれます。

現行のプログラム・ユニットが使用可能でない場合、コールのトレースでは、プログラム・ユニットの型、行番号およびスタックの深さが書き込まれます。

例外のトレースでは、行番号が書き込まれます。例外が発生すると、その例外がユーザー定義か事前定義のいずれであるかがトレースされ、事前定義の例外の場合は例外番号の情報がトレースされます。

# サブプログラムの要約

表 50-1 DBMS\_TRACE パッケージのサブプログラム

サブプログラム	説明
<a href="#">SET_PLSQL_TRACE プロシージャ</a> (50-4 ページ)	現在のセッションでトレースを開始します。
<a href="#">CLEAR_PLSQL_TRACE プロシージャ</a> (50-4 ページ)	セッションでのトレース・データのダンプを停止します。
<a href="#">PLSQL_TRACE_VERSION プロシージャ</a> (50-5 ページ)	トレース・パッケージのバージョン番号を取得します。

## SET\_PLSQL\_TRACE プロシージャ

このプロシージャは、PL/SQL のトレース・データ収集を可能にします。

### 構文

```
DBMS_TRACE.SET_PLSQL_TRACE (  
    trace_level INTEGER);
```

### パラメータ

表 50-2 SET\_PLSQL\_TRACE プロシージャのパラメータ

パラメータ	説明
trace_level	trace_all_calls、trace_enabled_calls、trace_all_exceptions または trace_enabled_exceptions のいずれか 1 つの定数を指定する必要があります。  詳細は、「 <a href="#">トレース・データの収集</a> 」(50-3 ページ)を参照してください。

## CLEAR\_PLSQL\_TRACE プロシージャ

このプロシージャは、トレース・データ収集を使用禁止にします。

### 構文

```
DBMS_TRACE.CLEAR_PLSQL_TRACE;
```

### パラメータ

なし。

# PLSQL\_TRACE\_VERSION プロシージャ

このプロシージャは、トレース・パッケージのバージョン番号を取得します。DBMS\_TRACE パッケージのバージョン番号とリリース番号を戻します。

## 構文

```
DBMS_TRACE.PLSQL_TRACE_VERSION (  
    major OUT BINARY_INTEGER,  
    minor OUT BINARY_INTEGER);
```

## パラメータ

表 50-3 PLSQL\_TRACE\_VERSION プロシージャのパラメータ

パラメータ	説明
major	DBMS_TRACE のバージョン番号。
minor	DBMS_TRACE のリリース番号。



---

## DBMS\_TRANSACTION

このパッケージは、ストアド・プロシージャから SQL トランザクション文へのアクセスを提供します。

**関連項目：**『Oracle8i SQL リファレンス』

要件

このパッケージは、パッケージ所有者 SYS ではなく、コール・ユーザーの権限で実行されま  
す。

サブプログラムの要約

表 51-1 DBMS\_TRANSACTION パッケージのサブプログラム

サブプログラム
<a href="#">READ_ONLY プロシージャ</a> ( 51-3 ページ )
<a href="#">READ_WRITE プロシージャ</a> ( 51-3 ページ )
<a href="#">ADVISE_ROLLBACK プロシージャ</a> ( 51-3 ページ )
<a href="#">ADVISE_NOTHING プロシージャ</a> ( 51-4 ページ )
<a href="#">ADVISE_COMMIT プロシージャ</a> ( 51-4 ページ )
<a href="#">USE_ROLLBACK_SEGMENT プロシージャ</a> ( 51-4 ページ )
<a href="#">COMMIT_COMMENT プロシージャ</a> ( 51-5 ページ )
<a href="#">COMMIT_FORCE プロシージャ</a> ( 51-5 ページ )
<a href="#">COMMIT プロシージャ</a> ( 51-6 ページ )
<a href="#">SAVEPOINT プロシージャ</a> ( 51-6 ページ )
<a href="#">ROLLBACK プロシージャ</a> ( 51-7 ページ )
<a href="#">ROLLBACK_SAVEPOINT プロシージャ</a> ( 51-7 ページ )
<a href="#">ROLLBACK_FORCE プロシージャ</a> ( 51-8 ページ )
<a href="#">BEGIN_DISCRETE_TRANSACTION プロシージャ</a> ( 51-8 ページ )
<a href="#">PURGE_MIXED プロシージャ</a> ( 51-9 ページ )
<a href="#">PURGE_LOST_DB_ENTRY プロシージャ</a> ( 51-10 ページ )
<a href="#">LOCAL_TRANSACTION_ID ファンクション</a> ( 51-12 ページ )
<a href="#">STEP_ID ファンクション</a> ( 51-12 ページ )

## READ\_ONLY プロシージャ

このプロシージャは、次の SQL 文と同じです。

```
SET TRANSACTION READ ONLY
```

### 構文

```
DBMS_TRANSACTION.READ_ONLY;
```

### パラメータ

なし。

## READ\_WRITE プロシージャ

このプロシージャは、次の SQL 文と同じです。

```
SET TRANSACTION READ WRITE
```

### 構文

```
DBMS_TRANSACTION.READ_WRITE;
```

### パラメータ

なし。

## ADVISE\_ROLLBACK プロシージャ

このプロシージャは、次の SQL 文と同じです。

```
ALTER SESSION ADVISE ROLLBACK
```

### 構文

```
DBMS_TRANSACTION.ADVISE_ROLLBACK;
```

### パラメータ

なし。

## ADVISE\_NOTHING プロシージャ

このプロシージャは、次の SQL 文と同じです。

```
ALTER SESSION ADVISE NOTHING
```

### 構文

```
DBMS_TRANSACTION.ADVISE_NOTHING;
```

### パラメータ

なし。

## ADVISE\_COMMIT プロシージャ

このプロシージャは、次の SQL 文と同じです。

```
ALTER SESSION ADVISE COMMIT
```

### 構文

```
DBMS_TRANSACTION.ADVISE_COMMIT;
```

### パラメータ

なし。

## USE\_ROLLBACK\_SEGMENT プロシージャ

このプロシージャは、次の SQL 文と同じです。

```
SET TRANSACTION USE ROLLBACK SEGMENT <rb_seg_name>
```

### 構文

```
DBMS_TRANSACTION.USE_ROLLBACK_SEGMENT (
    rb_name VARCHAR2);
```

### パラメータ

表 51-2 USE\_ROLLBACK\_SEGMENT プロシージャのパラメータ

パラメータ	説明
rb_name	使用するロールバック・セグメントの名前。



## COMMIT\_COMMENT プロシージャ

このプロシージャは、次の SQL 文と同じです。

```
COMMIT COMMENT <text>
```

### 構文

```
DBMS_TRANSACTION.COMMIT_COMMENT (  
    cmt VARCHAR2);
```

### パラメータ

表 51-3 COMMIT\_COMMENT プロシージャのパラメータ

パラメータ	説明
cmt	このコミットに関連するコメント。

## COMMIT\_FORCE プロシージャ

このプロシージャは、次の SQL 文と同じです。

```
COMMIT FORCE <text>, <number>"
```

### 構文

```
DBMS_TRANSACTION.COMMIT_FORCE (  
    xid VARCHAR2,  
    scn VARCHAR2 DEFAULT NULL);
```

### パラメータ

表 51-4 COMMIT\_FORCE プロシージャのパラメータ

パラメータ	説明
xid	ローカルまたはグローバルなトランザクション ID。
scn	システム変更番号。

## COMMIT プロシージャ

このプロシージャは、次の SQL 文と同じです。

```
COMMIT
```

このプロシージャは、PL/SQL の一部としてすでにインプリメントされています。

### 構文

```
DBMS_TRANSACTION.COMMIT;
```

### パラメータ

なし。

## SAVEPOINT プロシージャ

このプロシージャは、次の SQL 文と同じです。

```
SAVEPOINT <savepoint_name>
```

このプロシージャは、PL/SQL の一部としてすでにインプリメントされています。

### 構文

```
DBMS_TRANSACTION.SAVEPOINT (
    savept VARCHAR2);
```

### パラメータ

表 51-5 SAVEPOINT プロシージャのパラメータ

パラメータ	説明
savept	セーブポイントの識別子。

# ROLLBACK プロシージャ

このプロシージャは、次の SQL 文と同じです。

```
ROLLBACK
```

このプロシージャは、PL/SQL の一部としてすでにインプリメントされています。

## 構文

```
DBMS_TRANSACTION.ROLLBACK;
```

## パラメータ

なし。

# ROLLBACK\_SAVEPOINT プロシージャ

このプロシージャは、次の SQL 文と同じです。

```
ROLLBACK TO SAVEPOINT <savepoint_name>
```

このプロシージャは、PL/SQL の一部としてすでにインプリメントされています。

## 構文

```
DBMS_TRANSACTION.ROLLBACK_SAVEPOINT (  
    savept VARCHAR2);
```

## パラメータ

表 51-6 ROLLBACK\_SAVEPOINT プロシージャのパラメータ

パラメータ	説明
savept	セーブポイントの識別子。

## ROLLBACK\_FORCE プロシージャ

このプロシージャは、次の SQL 文と同じです。

```
ROLLBACK FORCE <text>
```

### 構文

```
DBMS_TRANSACTION.ROLLBACK_FORCE (  
    xid VARCHAR2);
```

### パラメータ

表 51-7 ROLLBACK\_FORCE プロシージャのパラメータ

パラメータ	説明
xid	ローカルまたはグローバルなトランザクション ID。

## BEGIN\_DISCRETE\_TRANSACTION プロシージャ

このプロシージャは、このトランザクションについて離散トランザクション・モードを設定します。

### 構文

```
DBMS_TRANSACTION.BEGIN_DISCRETE_TRANSACTION;
```

### パラメータ

なし。

### 例外

表 51-8 BEGIN\_DISCRETE\_TRANSACTION プロシージャの例外

例外	説明
ORA-08175	トランザクションが、離散トランザクションとして実行できない操作を行おうとしました。  この例外が発生した場合、ロールバックしてトランザクションを再試行します。
ORA-08176	トランザクションが、ロールバック・データを生成しない操作（索引の作成、ダイレクト・ロードまたは離散トランザクション）によって変更されたデータを検出しました。  この例外が発生した場合、例外を受け取った操作を再試行します。

例

```
DISCRETE_TRANSACTION_FAILED exception;
  pragma exception_init(DISCRETE_TRANSACTION_FAILED, -8175);
CONSISTENT_READ_FAILURE exception;
  pragma exception_init(CONSISTENT_READ_FAILURE, -8176);
```

PURGE\_MIXED プロシージャ

(自動回復で結果を解決せずに) インダウト・トランザクションを強制的にコミットまたはロールバックすると、そのトランザクションに混合出力を持つ可能性があります。つまり、一部のサイトはコミットされ、他はロールバックされます。このような矛盾は Oracle で自動的に解決できませんが、Oracle は、MIXED 列を値 'yes' に設定して、DBA\_2PC\_PENDING にあるエントリにフラグ付けを行います。

Oracle では、混合出力トランザクションの情報が自動的に削除されることはありません。混合トランザクションの結果として発生したすべての不整合が解決されたことを、アプリケーションまたは DBA で確認したら、このプロシージャを使用して、指定した混合出力トランザクションの情報を削除できます。

構文

```
DBMS_TRANSACTION.PURGE_MIXED (
  xid VARCHAR2);
```

パラメータ

表 51-9 PURGE\_MIXED プロシージャのパラメータ

パラメータ	説明
xid	DBA_2PC_PENDING 表にある LOCAL_TRAN_ID 列の値に設定する必要があります。

## PURGE\_LOST\_DB\_ENTRY プロシージャ

コミット処理中に障害が発生すると、自動回復機能によって、そのトランザクションに関連しているすべてのサイトでの結果が一貫して解決されます。ただし、回復の完了前にリモート・データベースが破損または再作成されると、DBA\_2PC\_PENDING 内の回復を制御するために使用するエントリと、それに関連する表が削除されずに、回復処理が定期的に再試行されます。プロシージャ PURGE\_LOST\_DB\_ENTRY では、このようなトランザクションをローカル・サイトから削除できます。

### 構文

```
DBMS_TRANSACTION.PURGE_LOST_DB_ENTRY (  
    xid VARCHAR2);
```

---

---

**警告：** PURGE\_LOST\_DB\_ENTRY は、他のデータベースが失われたり再作成された場合のみ使用してください。他の目的で使用すると、他のデータベースを回復不能にしたり一貫性のない状態にする可能性があります。

---

---

自動回復の実行前に、トランザクションは、DBA\_2PC\_PENDING を "collecting"、"committed" または "prepared" の状態で表示できます。DBA が "commit force" または "rollback force" を使用して、インダウト・トランザクションに特定の結果を強制した場合は、"forced commit" または "forced rollback" の状態も表示できます。自動回復は通常、このような状態のエントリを削除します。唯一の例外は、トランザクション内の他のサイトとの間で整合性がとられていない状態の強制トランザクションが回復処理で見つかった場合です。この場合、エントリは表内に残り、MIXED 列の値が 'yes' になります。

ただし、状態によっては、自動回復の実行が不可能な場合があります。たとえば、リモート・データベースが完全に破損した場合です。たとえ再作成しても、新規のデータベース ID を取得するので、回復処理では識別できません（起こり得るエラーは、ORA-02062）。このような場合、DBA はプロシージャ PURGE\_LOST\_DB\_ENTRY を使用して、"prepared" 以外の状態にあるエントリをクリーン・アップできます。これらのエントリはデータベース・リソースを保持していないため、DBA は特に急いでこれらのエントリを解決する必要はありません。

次の表は、トランザクションについてのさまざまな状態の内容および DBA がとるべきアクションを示します。

表 51-10 PURGE\_LOST\_DB\_ENTRY プロシージャの状態

列の状態	グローバル・ トランザク ションの状態	ローカル・ トランザク ションの状態	通常の DBA アクション	代替の DBA アクション
Collecting	ロールバック	ロールバック	なし	PURGE_LOST_DB_ENTRY (1)
Committed	コミット済	コミット済	なし	PURGE_LOST_DB_ENTRY (1)
Prepared	不明	準備済	なし	FORCE COMMIT または ROLLBACK
Forced commit	不明	コミット済	なし	PURGE_LOST_DB_ENTRY (1)
Forced rollback	不明	ロールバック	なし	PURGE_LOST_DB_ENTRY (1)
Forced commit (mixed)	混合	コミット済	(2)	
Forced rollback (mixed)	混合	ロールバック	(2)	

**注意 1:** 重要な再構成が発生して自動回復ではトランザクションを解決できない場合のみ使用します。例としては、リモート・データベース全体が破損し、ソフトウェアでの再構成によって 2 フェーズ・コミット機能を失う結果になった場合や TP モニターのような外部トランザクション・コーディネータからの情報を失った場合などがあります。

**注意 2:** 不整合を取り除くために調査したり手動で作業を行い、プロシージャ PURGE\_MIXED を使用します。

## パラメータ

表 51-11 PURGE\_LOST\_DB\_ENTRY プロシージャのパラメータ

パラメータ	説明
xid	DBA_2PC_PENDING 表にある LOCAL_TRAN_ID 列の値に設定する必要があります。

## LOCAL\_TRANSACTION\_ID ファンクション

このファンクションは、現行のトランザクションについてローカルな（インスタンスに対して）一意の識別子を戻します。現行のトランザクションがない場合は、NULL を戻します。

### 構文

```
DBMS_TRANSACTION.LOCAL_TRANSACTION_ID (  
    create_transaction BOOLEAN := FALSE)  
RETURN VARCHAR2;
```

### パラメータ

表 51-12 LOCAL\_TRANSACTION\_ID ファンクションのパラメータ

パラメータ	説明
create_transaction	TRUE に設定すると、トランザクションが現在アクティブでない場合は、トランザクションを開始します。

## STEP\_ID ファンクション

このファンクションは、トランザクションの DML 操作を順序付ける、ローカルで（ローカル・トランザクションに対して）一意の正の整数を戻します。

### 構文

```
DBMS_TRANSACTION.STEP_ID  
RETURN NUMBER;
```

### パラメータ

なし。



このパッケージは、トランスポート可能なセットが自己完結型かどうかをチェックします。すべての違反が、ビュー `TRANSPORT_SET_VIOLATIONS` から選択できる一時表に挿入されます。

**関連項目：**『Oracle8i 管理者ガイド』および『Oracle8i 移行ガイド』

例外

```
ts_not_found EXCEPTION;
PRAGMA exception_init(ts_not_found, -29304);
ts_not_found_num NUMBER := -29304;

invalid_ts_list EXCEPTION;
PRAGMA exception_init(invalid_ts_list, -29346);
invalid_ts_list_num NUMBER := -29346;

sys_or_tmp_ts EXCEPTION;
PRAGMA exception_init(sys_or_tmp_ts, -29351);
sys_or_tmp_ts_num NUMBER := -29351;
```

サブプログラムの要約

この 2 つのプロシージャは、データベース管理者がコールするように設計されています。

表 52-1 DBMS\_TTS パッケージのサブプログラム

サブプログラム	説明
<a href="#">TRANSPORT_SET_CHECK プロシージャ</a> ( 52-2 ページ )	表領域 ( トランスポート可能 ) のセットが自己完結型かどうかをチェックします。
<a href="#">DOWNGRADE プロシージャ</a> ( 52-3 ページ )	データに関連するトランスポート可能な表領域をダウングレードします。

TRANSPORT\_SET\_CHECK プロシージャ

このプロシージャは、表領域 ( トランスポート可能 ) のセットが自己完結型かどうかをチェックします。このプロシージャのコール後、ユーザーはビューから選択して、違反があればそのリストを調べることができます。ビューで行が戻らない場合、表領域のセットは自己完結型です。次に例を示します。

```
SVRMGR> EXECUTE TRANSPORT_SET_CHECK('foo,bar', TRUE);
SVRMGR> SELECT * FROM TRANSPORT_SET_VIOLATIONS;
```

構文

```
DBMS_TTS.TRANSPORT_SET_CHECK (
    ts_list          IN VARCHAR2,
    incl_constraints IN BOOLEAN);
```

パラメータ

表 52-2 TRANSPORT\_SET\_CHECK プロシージャのパラメータ

パラメータ	説明
ts_list	表領域のカンマで区切られたリスト。
incl_constraints	表領域が自己完結型かどうかを調べるときに、参照整合性制約を考慮する場合は TRUE を設定します。

DOWNGRADE プロシージャ

このプロシージャは、データに関連するトランスポート可能な表領域をダウングレードします。

構文

```
DBMS_TTS.DOWNGRADE;
```

パラメータ

なし。



---

## DBMS\_UTILITY

このパッケージは、各種のユーティリティ・サブプログラムを提供します。

DBMS\_UTILITY は、各パーティションに対してジョブを実行します。INIT.ORA パラメータの JOB\_QUEUE\_PROCESSES を正しく設定して、同時実行ジョブの数を制御するのは、ユーザーの責任です。正しい構文についての最小限度のエラー・チェックがあります。すべてのエラーは、SNP トレース・ファイルにレポートされます。

要件

DBMS\_UTILITY は、NAME\_RESOLVE、COMPILE\_SCHEMA および ANALYZE\_SCHEMA プロシージャに対するコール・ユーザーの権限で実行されます。これは、SQL の正しい動作のために必要です。

このパッケージは、SYS として実行されません。権限は、DBMS\_DDL を介してチェックされます。

型

type uncl\_array IS TABLE OF VARCHAR2(227) INDEX BY BINARY\_INTEGER;  
"USER"."NAME"."COLUMN"@LINK のリストは、ここに格納されます。

type name\_array IS TABLE OF VARCHAR2(30) INDEX BY BINARY\_INTEGER;  
NAME のリストは、ここに格納されます。

type dblink\_array IS TABLE OF VARCHAR2(128) INDEX BY BINARY\_INTEGER;  
データベース・リンクのリストは、ここに格納されます。

TYPE index\_table\_type IS TABLE OF BINARY\_INTEGER INDEX BY BINARY\_INTEGER;  
オブジェクトの生成順序は、ここに戻されます。

TYPE number\_array IS TABLE OF NUMBER INDEX BY BINARY\_INTEGER;  
ユーザーのためのオブジェクトの生成順序は、ここに戻されます。

TYPE instance\_record IS RECORD (  
    inst\_number    NUMBER,  
    inst\_name      VARCHAR2(60));  
TYPE instance\_table IS TABLE OF instance\_record INDEX BY BINARY\_INTEGER;  
アクティブなインスタンス番号とインスタンス名のリスト。

instance\_table の索引の開始は 1 で、instance\_table は濃密です。

サブプログラムの要約

表 53-1 DBMS\_UTILITY パッケージのサブプログラム ( 1 / 3 ページ)

サブプログラム	説明
<a href="#">COMPILE_SCHEMA プロシージャ</a> ( 53-4 ページ )	指定したスキーマ内にあるすべてのプロシージャ、ファンクション、パッケージおよびトリガーをコンパイルします。
<a href="#">ANALYZE_SCHEMA プロシージャ</a> ( 53-5 ページ )	スキーマ内にあるすべての表、クラスタおよび索引を分析します。
<a href="#">ANALYZE_DATABASE プロシージャ</a> ( 53-6 ページ )	データベース内にあるすべての表、クラスタおよび索引を分析します。

表 53-1 DBMS\_UTILITY パッケージのサブプログラム ( 2 / 3 ページ)

サブプログラム	説明
<a href="#">FORMAT_ERROR_STACK</a> ファンクション ( 53-7 ページ )	現行のエラー・スタックをフォーマットします。
<a href="#">FORMAT_CALL_STACK</a> ファンクション ( 53-7 ページ )	現行の呼出しスタックをフォーマットします。
<a href="#">IS_PARALLEL_SERVER</a> ファンクション ( 53-8 ページ )	このデータベースがパラレル・サーバー・モードで実行しているかどうかを検出します。
<a href="#">GET_TIME</a> ファンクション ( 53-8 ページ )	現在の時間を 100 分の 1 秒単位で検出します。
<a href="#">GET_PARAMETER_VALUE</a> ファンクシ ン ( 53-9 ページ )	指定した init.ora パラメータの値を取得します。
<a href="#">NAME_RESOLVE</a> プロシージャ ( 53-10 ページ )	指定した名前を解決します。
<a href="#">NAME_TOKENIZE</a> プロシージャ ( 53-12 ページ )	指定した名前を解析するために解析機能をコールします。
<a href="#">COMMA_TO_TABLE</a> プロシージャ ( 53-12 ページ )	名前のカンマで区切られたリストを、PL/SQL 名前表に変換します。
<a href="#">TABLE_TO_COMMA</a> プロシージャ ( 53-13 ページ )	PL/SQL 名前表を、名前のカンマで区切られたリストに変換します。
<a href="#">PORT_STRING</a> ファンクション ( 53-14 ページ )	Oracle とオペレーティング・システムのバージョンを一 意に識別する文字列を戻します。
<a href="#">DB_VERSION</a> プロシージャ ( 53-14 ページ )	データベースに関するバージョン情報を戻します。
<a href="#">MAKE_DATA_BLOCK_ADDRESS</a> ファンク ション ( 53-15 ページ )	ファイル番号とブロック番号を指定して、データ・ブロッ ク・アドレスを作成します。
<a href="#">DATA_BLOCK_ADDRESS_FILE</a> ファンク ション ( 53-15 ページ )	データ・ブロック・アドレスのファイル番号部分を取得し ます。
<a href="#">DATA_BLOCK_ADDRESS_BLOCK</a> ファン クション ( 53-16 ページ )	データ・ブロック・アドレスのブロック番号部分を取得し ます。
<a href="#">GET_HASH_VALUE</a> ファンクション ( 53-17 ページ )	指定した文字列についてハッシュ値を計算します。
<a href="#">ANALYZE_PART_OBJECT</a> プロシージャ ( 53-18 ページ )	オブジェクトの各パーティションを並列に実行します。
<a href="#">EXEC_DDL_STATEMENT</a> プロシージャ ( 53-19 ページ )	parse_string で DDL 文を実行します。

表 53-1 DBMS\_UTILITY パッケージのサブプログラム ( 3 / 3 ページ)

サブプログラム	説明
<a href="#">CURRENT_INSTANCE</a> ファンクション ( 53-19 ページ )	現在接続しているインスタンス番号を戻します。
<a href="#">ACTIVE_INSTANCES</a> プロシージャ ( 53-19 ページ )	現在接続しているインスタンス番号と名前のリストおよびインスタンスの数を戻します。

COMPILE\_SCHEMA プロシージャ

このプロシージャは、指定したスキーマ内にあるすべてのプロシージャ、ファンクション、パッケージおよびトリガーをコンパイルします。このプロシージャのコール後、ステータスが INVALID の項目をビュー ALL\_OBJECTS から選択して、すべてのオブジェクトが正常にコンパイルされたかどうかを調べます。

Enterprise Manager のコマンドを使用すると、INVALID のオブジェクトに関連するエラーを表示できます。

```
SHOW ERRORS <type> <schema>.<name>
```

構文

```
DBMS_UTILITY.COMPILE_SCHEMA (  
    schema VARCHAR2);
```

パラメータ

表 53-2 COMPILE\_SCHEMA プロシージャのパラメータ

パラメータ	説明
schema	スキーマ名。

例外

表 53-3 COMPILE\_SCHEMA プロシージャの例外

例外	説明
ORA-20000	このスキーマ内のいずれかのオブジェクトに対して権限が不十分です。



# ANALYZE\_SCHEMA プロシージャ

このプロシージャは、スキーマ内にあるすべての表、クラスタおよび索引を分析します。

## 構文

```
DBMS_UTILITY.ANALYZE_SCHEMA (  
    schema          VARCHAR2,  
    method          VARCHAR2,  
    estimate_rows   NUMBER   DEFAULT NULL,  
    estimate_percent NUMBER   DEFAULT NULL,  
    method_opt      VARCHAR2 DEFAULT NULL);
```

## パラメータ

表 53-4 ANALYZE\_SCHEMA プロシージャのパラメータ

パラメータ	説明
schema	スキーマ名。
method	ESTIMATE、COMPUTE または DELETE のいずれかを設定します。  ESTIMATE の場合は、estimate_rows または estimate_percent のどちらかをゼロ以外に設定する必要があります。
estimate_rows	推定する行数。
estimate_percent	推定する行のパーセント。  estimate_rows が指定されている場合、このパラメータは無視されます。
method_opt	次の書式の分析方法オプション。  [ FOR TABLE ] [ FOR ALL [INDEXED] COLUMNS] [SIZE n] [ FOR ALL INDEXES ]

## 例外

表 53-5 ANALYZE\_SCHEMA プロシージャの例外

例外	説明
ORA-20000	このスキーマ内のいずれかのオブジェクトに対して権限が不十分です。

## ANALYZE\_DATABASE プロシージャ

このプロシージャは、データベース内にあるすべての表、クラスタおよび索引を分析します。

### 構文

```
DBMS_UTILITY.ANALYZE_DATABASE (  
    method          VARCHAR2,  
    estimate_rows    NUMBER    DEFAULT NULL,  
    estimate_percent NUMBER    DEFAULT NULL,  
    method_opt       VARCHAR2 DEFAULT NULL);
```

### パラメータ

表 53-6 ANALYZE\_DATABASE プロシージャのパラメータ

パラメータ	説明
method	ESTIMATE、COMPUTE または DELETE のいずれかを設定します。  ESTIMATE の場合は、estimate_rows または estimate_percent のどちらかをゼロ以外に設定する必要があります。
estimate_rows	推定する行数。
estimate_percent	推定する行のパーセント。  estimate_rows が指定されている場合、このパラメータは無視されます。
method_opt	次の書式の分析方法オプション。  [ FOR TABLE ] [ FOR ALL [INDEXED] COLUMNS] [SIZE n] [ FOR ALL INDEXES ]

### 例外

表 53-7 ANALYZE\_DATABASE プロシージャの例外

例外	説明
ORA-20000	このデータベース内のいずれかのオブジェクトに対して権限が不十分です。

## FORMAT\_ERROR\_STACK ファンクション

このファンクションは、現行のエラー・スタックをフォーマットします。このファンクションは、全エラー・スタックを表示するための例外ハンドラで使用できます。

### 構文

```
DBMS_UTILITY.FORMAT_ERROR_STACK  
RETURN VARCHAR2;
```

### パラメータ

なし。

### 戻り値

最大 2000 バイトまでのエラー・スタックを戻します。

## FORMAT\_CALL\_STACK ファンクション

このファンクションは、現行の呼出しスタックをフォーマットします。このファンクションは、呼出しスタックにアクセスするための任意ストアド・プロシージャまたはトリガーで使用できます。このファンクションは、デバッグ時に役立ちます。

### 構文

```
DBMS_UTILITY.FORMAT_CALL_STACK  
RETURN VARCHAR2;
```

### パラメータ

なし。

### プラグマ

```
pragma restrict_references(format_call_stack,WNDS);
```

### 戻り値

最大 2000 バイトまでの呼出しスタックを戻します。

## IS\_PARALLEL\_SERVER ファンクション

このファンクションは、このデータベースがパラレル・サーバー・モードで実行しているかどうかを検出します。

### 構文

```
DBMS_UTILITY.IS_PARALLEL_SERVER  
RETURN BOOLEAN;
```

### パラメータ

なし。

### 戻り値

インスタンスがパラレル・サーバー・モードで起動されている場合は TRUE を戻し、そうでない場合は FALSE を戻します。

## GET\_TIME ファンクション

このファンクションは、現在の時間を 100 分の 1 秒単位で検出します。これは、主に経過時間を判断するのに役立ちます。

### 構文

```
DBMS_UTILITY.GET_TIME  
RETURN NUMBER;
```

### パラメータ

なし。

### 戻り値

任意の時点からの時間を 100 分の 1 秒の数で戻します。

## GET\_PARAMETER\_VALUE ファンクション

このファンクションは、指定した `init.ora` パラメータの値を取得します。

### 構文

```
DBMS_UTILITY.GET_PARAMETER_VALUE (  
    parnam IN      VARCHAR2,  
    intval IN OUT  BINARY_INTEGER,  
    strval IN OUT  VARCHAR2)  
RETURN BINARY_INTEGER;
```

### パラメータ

表 53-8 GET\_PARAMETER\_VALUE ファンクションのパラメータ

パラメータ	説明
parnam	パラメータ名。
intval	整数パラメータの値、または文字列パラメータの値の長さ。
strval	文字列パラメータの値。

### 戻り値

表 53-9 GET\_PARAMETER\_VALUE ファンクションの戻り値

戻り値	説明
partyp	パラメータ型。 パラメータが整数またはブール・パラメータの場合は 0。 パラメータが文字列またはファイル・パラメータの場合は 1。

### 例

```
DECLARE  
    parnam VARCHAR2(256);  
    intval BINARY_INTEGER;  
    strval VARCHAR2(256);  
    partyp BINARY_INTEGER;  
BEGIN  
    partyp := dbms_utility.get_parameter_value('max_dump_file_size',  
                                              intval, strval);  
    dbms_output.put('parameter value is: ');  
    IF partyp = 1 THEN
```

```
        dbms_output.put_line(strval);
ELSE
    dbms_output.put_line(intval);
END IF;
IF partyp = 1 THEN
    dbms_output.put('parameter value length is: ');
    dbms_output.put_line(intval);
END IF;
dbms_output.put('parameter type is: ');
IF partyp = 1 THEN
    dbms_output.put_line('string');
ELSE
    dbms_output.put_line('integer');
END IF;
END;
```

## NAME\_RESOLVE プロシージャ

このプロシージャは、指定した名前を解決し、必要に応じてシノニム変換と認可チェックが含まれます。

### 構文

```
DBMS_UTILITY.NAME_RESOLVE (
    name           IN  VARCHAR2,
    context        IN  NUMBER,
    schema         OUT VARCHAR2,
    part1          OUT VARCHAR2,
    part2          OUT VARCHAR2,
    dblink         OUT VARCHAR2,
    part1_type     OUT NUMBER,
    object_number  OUT NUMBER);
```

## パラメータ

表 53-10 NAME\_RESOLVE プロシージャのパラメータ

パラメータ	説明
name	<p>オブジェクト名。</p> <p>この名前はフォーム [[a.]b.]c[@d] で指定します。ここで、a、b、c は SQL 識別子、d は DB リンクです。DB リンクでは、構文チェックは実行されません。DB リンクが指定されている場合や名前が DB リンクの一部に変換される場合、オブジェクトは解決されませんが、schema、part1、part2 および dblink OUT の各パラメータは入力されます。</p> <p>a、b および c は、デリミタ付き識別子の場合があり、NLS 文字（シングルおよびマルチバイト）を含んでいる場合があります。</p>
context	0 ~ 8 の範囲の整数を指定します。
schema	オブジェクト c のスキーマ。name パラメータにスキーマが指定されていない場合、schema は、名前を解決して決定されます。
part1	名前の最初の部分。この名前の型は、part1_type に指定されます（シノニム、プロシージャまたはパッケージ）。
part2	このパラメータが NULL 以外の場合は、part1 で示されるパッケージ内のプロシージャ名です。
dblink	このパラメータが NULL 以外の場合、データベース・リンクは、name の一部として指定されたか、または name がデータベース・リンクの一部に変換されるシノニムとして指定されたかのいずれかです。後者の場合、part1_type はシノニムを示します。
part1_type	<p>part1 の型は、次のとおりです。</p> <p>5 シノニム</p> <p>7 プロシージャ（最上位レベル）</p> <p>8 ファンクション（最上位レベル）</p> <p>9 パッケージ</p> <p>シノニムの場合、name は、データベース・リンクの一部に変換するシノニムです。ここで、名前変換がさらに必要な場合は、このリモート・ノードで DBMS_UTILITY.NAME_RESOLVE プロシージャをコールする必要があります。</p>

## 例外

すべてのエラーは、例外を呼び出すことによって処理されます。オブジェクト名の指定時に起こり得る各種の構文エラーに基づいて、広範囲にわたる例外が用意されています。

## NAME\_TOKENIZE プロシージャ

このプロシージャは、解析機能をコールして、"a [. b [. c ]][@ dblink ]" と指定した名前を解析します。二重引用符を削除するか、または引用符がない場合は大文字に変換します。ソートに関するすべてのコメントは無視し、意味的な分析は行いません。不明な値は NULL のまま残ります。

### 構文

```
DBMS_UTILITY.NAME_TOKENIZE (  
    name      IN  VARCHAR2,  
    a         OUT VARCHAR2,  
    b         OUT VARCHAR2,  
    c         OUT VARCHAR2,  
    dblink    OUT VARCHAR2,  
    nextpos   OUT BINARY_INTEGER);
```

### パラメータ

各 a、b、c および dblink は、それぞれ次の anext、bnext、cnext、dnext の各トークンが開始される場所を示します。

## COMMA\_TO\_TABLE プロシージャ

このプロシージャは、名前のカンマで区切られたリストを、PL/SQL 名前表に変換します。このプロシージャは、NAME\_TOKENIZE を使用して、名前とカンマを区別します。

### 構文

```
DBMS_UTILITY.COMMA_TO_TABLE (  
    list      IN  VARCHAR2,  
    tablen    OUT BINARY_INTEGER,  
    tab       OUT UNCL_ARRAY);
```

### パラメータ

表 53-11 COMMA\_TO\_TABLE プロシージャのパラメータ

パラメータ	説明
list	表のカンマで区切られたリスト。
tablen	PL/SQL 表にある表の数。
tab	表名のリストを含んだ PL/SQL 表。



戻り値

PL/SQL 表が、値 1..n と n+1 is null とともに戻されます。

使用上の注意

list は、空ではないカンマで区切られたリストである必要があります。カンマで区切られたリスト以外の場合は拒否されます。二重引用符内のカンマは無視されます。

tab にある値は、変換されずに元のリストからカットされます。

TABLE\_TO\_COMMA プロシージャ

このプロシージャは、PL/SQL 名前表を、名前のカンマで区切られたリストに変換します。PL/SQL 表を 1..n に変換して n+1 null で終了します。

構文

```
DBMS_UTILITY.TABLE_TO_COMMA (  
    tab      IN  UNCL_ARRAY,  
    tablen   OUT BINARY_INTEGER,  
    list     OUT VARCHAR2);
```

パラメータ

表 53-12 TABLE\_TO\_COMMA プロシージャのパラメータ

パラメータ	説明
tab	表名のリストを含んだ PL/SQL 表。
tablen	PL/SQL 表にある表の数。
list	表のカンマで区切られたリスト。

戻り値

カンマで区切られたリストと表内で検出された要素の数が戻されます (n), '',' || '',' || '',' = '','' に注意してください。

## PORT\_STRING ファンクション

このファンクションは、オペレーティング・システムと、TWO TASK PROTOCOL バージョンのデータベースを識別する文字列を戻します。たとえば、"VAX/VMX-7.1.0.0" が戻ります。

最大長はポート固有の長さです。

### 構文

```
DBMS_UTILITY.PORT_STRING
    RETURN VARCHAR2;
```

### パラメータ

なし。

### プラグマ

```
pragma restrict_references(port_string, WNDS, RNDS, WNPS, RNPS);
```

## DB\_VERSION プロシージャ

このプロシージャは、データベースに関するバージョン情報を戻します。

### 構文

```
DBMS_UTILITY.DB_VERSION (
    version          OUT VARCHAR2,
    compatibility OUT VARCHAR2);
```

### パラメータ

表 53-13 DB\_VERSION プロシージャのパラメータ

パラメータ	説明
version	データベースの内部ソフトウェア・バージョンを表す文字列 (例 : 7.1.0.0.0 )  この文字列の長さは可変なので、データベース・バージョンによって決定されます。
compatibility	互換性がある init.ora パラメータによって決定する、データベースの互換性設定。  このパラメータが init.ora ファイルで指定されていない場合は、NULL が戻されます。

## MAKE\_DATA\_BLOCK\_ADDRESS ファンクション

このファンクションは、ファイル番号とブロック番号を指定したデータ・ブロック・アドレスを作成します。データ・ブロック・アドレスは、データベース内のブロックを識別するために使用する内部構造です。このファンクションは、データ・ブロック・アドレスを含んだ特定の固定表にアクセスするとき役立ちます。

### 構文

```
DBMS_UTILITY.MAKE_DATA_BLOCK_ADDRESS (  
    file NUMBER,  
    block NUMBER)  
RETURN NUMBER;
```

### パラメータ

表 53-14 MAKE\_DATA\_BLOCK\_ADDRESS ファンクションのパラメータ

パラメータ	説明
file	ブロックを含んだファイル。
block	ブロック増分値に基づくファイル内でのブロックのオフセット。

### プラグマ

```
pragma restrict_references(make_data_block_address, WNDS, RNDS, WNPS, RNPS);
```

### 戻り値

表 53-15 MAKE\_DATA\_BLOCK\_ADDRESS ファンクションの戻り値

戻り値	説明
dba	データ・ブロック・アドレス。

## DATA\_BLOCK\_ADDRESS\_FILE ファンクション

このファンクションは、データ・ブロック・アドレスのファイル番号部分を取得します。

### 構文

```
DBMS_UTILITY.DATA_BLOCK_ADDRESS_FILE (  
    dba NUMBER)  
RETURN NUMBER;
```

パラメータ

表 53-16 DATA\_BLOCK\_ADDRESS\_FILE ファンクションのパラメータ

パラメータ	説明
dba	データ・ブロック・アドレス。

プラグマ

```
pragma restrict_references(data_block_address_file, WNDS, RNDS, WNPS, RNPS);
```

戻り値

表 53-17 DATA\_BLOCK\_ADDRESS\_FILE ファンクションの戻り値

戻り値	説明
file	ブロックを含むファイル。

DATA\_BLOCK\_ADDRESS\_BLOCK ファンクション

このファンクションは、データ・ブロック・アドレスのブロック番号部分を取得します。

構文

```
DBMS_UTILITY.DATA_BLOCK_ADDRESS_BLOCK (  
    dba NUMBER)  
    RETURN NUMBER;
```

パラメータ

表 53-18 DATA\_BLOCK\_ADDRESS\_BLOCK ファンクションのパラメータ

パラメータ	説明
dba	データ・ブロック・アドレス。

プラグマ

```
pragma restrict_references(data_block_address_block, WNDS, RNDS, WNPS, RNPS);
```

## 戻り値

表 53-19 DATA\_BLOCK\_ADDRESS\_BLOCK ファンクションの戻り値

戻り値	説明
block	ブロックのブロック・オフセット。

## GET\_HASH\_VALUE ファンクション

このファンクションは、指定した文字列についてハッシュ値を計算します。

### 構文

```
DBMS_UTILITY.GET_HASH_VALUE (  
    name          VARCHAR2,  
    base          NUMBER,  
    hash_size     NUMBER)  
RETURN NUMBER;
```

### パラメータ

表 53-20 GET\_HASH\_VALUE ファンクションのパラメータ

パラメータ	説明
name	ハッシュする文字列。
base	戻されるハッシュ値が始まる基礎値。
hash_size	必要とするハッシュ表のサイズ。

### プラグマ

```
pragma restrict_references(get_hash_value, WNDS, RNDS, WNPS, RNPS);
```

### 戻り値

ハッシュ値は、入力文字列に基づいています。たとえば、ハッシュ値が 1000 ~ 3047 の範囲にある文字列についてハッシュ値を取得するには、基礎値として 1000、hash\_size 値として 2048 を使用します。hash\_size パラメータは、2 の累乗を使用すると動作が最適になります。

## ANALYZE\_PART\_OBJECT プロシージャ

このプロシージャは、次の SQL と同じです。

```
ANALYZE TABLE|INDEX [<schema>.<object_name> PARTITION <pname> [<command_type>]
[<command_opt>] [<sample_clause>]
```

オブジェクトの各パーティションについて、ジョブ・キューを使用して並列に実行します。

### 構文

```
DBMS_UTILITY.ANALYZE_PART_OBJECT (
    schema          IN VARCHAR2 DEFAULT NULL,
    object_name     IN VARCHAR2 DEFAULT NULL,
    object_type     IN CHAR      DEFAULT 'T',
    command_type    IN CHAR      DEFAULT 'E',
    command_opt     IN VARCHAR2 DEFAULT NULL,
    sample_clause   IN VARCHAR2 DEFAULT 'SAMPLE 5 PERCENT');
```

### パラメータ

表 53-21 ANALYZE\_PART\_OBJECT プロシージャのパラメータ

パラメータ	説明
schema	object_name のスキーマ。
object_name	分析するオブジェクトの名前。パーティション化されている必要があります。
object_type	オブジェクトの型。T (表) または I (索引) である必要があります。
command_type	次のいずれかになります。 C (統計情報の計算) E (統計情報の推定) D (統計情報の削除) V (構造の検証)
command_opt	command_type のその他のオプション。 C と E については、FOR 表、FOR のすべての LOCAL 索引、FOR のすべての列、または分析統計 (表) の 'FOR' オプションをいくつか組み合わせることが可能です。V については、object_type が T のときに CASCADE が可能です。
sample_clause	command_type が 'E' の場合に使用するサンプル句。

## EXEC\_DDL\_STATEMENT プロシージャ

このプロシージャは、`parse_string` で DDL 文を実行します。

### 構文

```
DBMS_UTILITY.EXEC_DDL_STATEMENT (  
    parse_string IN VARCHAR2);
```

### パラメータ

表 53-22 EXEC\_DDL\_STATEMENT プロシージャのパラメータ

パラメータ	説明
<code>parse_string</code>	実行する DDL 文。

## CURRENT\_INSTANCE ファンクション

このファンクションは、現在接続しているインスタンス番号を戻します。接続しているインスタンスが切断されると、`NULL` を戻します。

### 構文

```
DBMS_UTILITY.CURRENT_INSTANCE  
    RETURN NUMBER;
```

### パラメータ

なし。

## ACTIVE\_INSTANCES プロシージャ

### 構文

```
DBMS_UTILITY.ACTIVE_INSTANCE (  
    instance_table OUT INSTANCE_TABLE,  
    instance_count OUT NUMBER);
```

## パラメータ

表 53-23 ACTIVE\_INSTANCES プロシージャのパラメータ

プロシージャ	説明
instance_table	アクティブなインスタンス番号と名前のリストが含まれます。進行中のインスタンスがない場合（または OPS 以外の設定の場合）リストは空になります。
instance_count	アクティブなインスタンスの数。OPS 以外の設定の場合は 0 になります。



---

## DEBUG\_EXTPROC

DEBUG\_EXTPROC パッケージによって、セッション内の extproc エージェントを起動できます。このユーティリティ・パッケージは、外部プロシージャをデバッグするのに役立ちます。

---

## 要件

Oracle アカウントに、パッケージに対する EXECUTE 権限と、CREATE LIBRARY 権限が必要です。

---

**注意：** DEBUG\_EXTPROC は、実行プロセスに連結できるデバッガを使用して、プラットフォームでのみ稼動します。

---

## インストール時の注意

パッケージをインストールするためには、スクリプト DBGEXTP.SQL を実行します。

- 'extproc' プロセスをデバッグする Oracle USER に、このパッケージをインストールまたはロードします。
- DEBUG\_EXTPROC パッケージに対する EXECUTE 権限があることを確認します。

```
SELECT SUBSTR(OBJECT_NAME, 1, 20)
FROM USER_OBJECTS
WHERE OBJECT_NAME = 'DEBUG_EXTPROC';
```

- パッケージに対して EXECUTE 権限がある場合は、他のユーザーとしてこのパッケージをインストールできます。

## DEBUG\_EXTPROC の使用方法

### 使用の前提条件

外部プロシージャ 'extproc' エージェントを起動するためには、リスナーが適切に構成されていることが前提です。

また、デバッグ処理を支援するために、デバッグ記号が付いた共有ライブラリを作成していることも前提です。C コンパイラのマニュアルをチェックして、適切な C コンパイラ・スイッチで、デバッグ記号が付いた共有ライブラリを作成してください。

### 使用上の注意

- Oracle に接続して、SQL\*Plus または OCI プログラムから新規の Oracle セッションを起動します。
- プロシージャ DEBUG\_EXTPROC.STARTUP\_EXTPROC\_AGENT を実行して、このセッションで extproc エージェントを起動し、たとえば、DEBUG\_EXTPROC.STARTUP\_EXTPROC\_AGENT を実行します。Extproc エージェントが終了してしまうので、このセッションは終了しないでください。
- このセッションで起動した extproc エージェントの PID を判別します。

- デバッガ（たとえば、gdb、dbx またはシステム固有のデバッガ）を使用して、extproc 実行ファイルをロードし、実行プロセスに連結します。
- ファンクション 'pextproc' にブレーク・ポイントを設定し、デバッガが実行を継続できるようにします。
- 最初に `DEBUG_EXTPROC.STARTUP_EXTPROC_AGENT` を実行した同じセッションで、外部プロシーダを実行します。
- デバッガがファンクション 'pextproc' でブレークします。この時点で、PL/SQL 外部ファンクションで参照する共有ライブラリがロードされ、ファンクションが解決されます。C ファンクションにブレーク・ポイントを設定し、デバッガが実行を継続できるようにします。

PL/SQL は、実行時に共有ライブラリをロードするため、使用するデバッガは、共有ライブラリから新規の記号を自動的に追跡管理できる場合とできない場合があります。デバッガ・コマンドをいくつか発行して、記号をロードできます（たとえば、gdb 内の 'share'）。

- デバッガは、C ファンクションでブレークします。デバッグ記号が付いた共有ライブラリを作成しておくことが前提です。
- デバッグを続行します。

## サブプログラムの要約

`DEBUG_EXTPROC` には、1 つのサブプログラムである `STARTUP_EXTPROC_AGENT` プロシーダが含まれています。これにより、セッション内で extproc エージェント・プロセスを起動します。

### STARTUP\_EXTPROC\_AGENT プロシーダ

このプロシーダは、セッションで extproc エージェント・プロセスを起動します。これにより、実行プロセスの PID を取得できます。この PID は、デバッガを使用して実行プロセスに連結するために必要です。

#### 構文

```
DEBUG_EXTPROC.STARTUP_EXTPROC_AGENT;
```

#### パラメータ

なし。



OUTLN\_PKG パッケージには、ストアド・アウトラインの管理に関連するサブプログラムのための機能インタフェースが含まれています。

ストアド・アウトラインは、指定した SQL 文についての実行プランに関連するストアド・データです。これによって、オプティマイザは、最初にアウトラインとともに生成されたプランと同じ実行プランを繰り返し再作成できます。アウトラインに格納されたデータの一部分は、プランの安定性を保つために使用するヒントで構成されています。

要件

OUTLN\_PKG には、適切なユーザーのみ使用できる管理プロシージャが含まれています。  
EXECUTE 権限は、DBA が明示的に定めない限り、一般ユーザー・コミュニティにまで拡張されません。

セキュリティ

アウトラインの管理目的に使用可能な PL/SQL ファンクションは、そのプロシージャ（またはパッケージ）に対する EXECUTE 権限があるユーザーのみが実行できます。

サブプログラムの要約

表 55-1 OUTLN\_PKG パッケージのサブプログラム

サブプログラム	説明
<a href="#">DROP_UNUSED プロシージャ</a> (55-2 ページ)	作成後に使用されなくなったアウトラインをすべて削除します。
<a href="#">DROP_BY_CAT プロシージャ</a> (55-3 ページ)	特定のカテゴリに属しているすべてのアウトラインを削除します。
<a href="#">UPDATE_BY_CAT プロシージャ</a> (55-3 ページ)	あるカテゴリにあるすべてのアウトラインのカテゴリを、新規のカテゴリに変更します。

DROP\_UNUSED プロシージャ

このプロシージャは、SQL 文のコンパイルで適用されなくなったアウトラインを削除します。

構文

```
OUTLN_PKG.DROP_UNUSED;
```

パラメータ

なし。

使用上の注意

動的 SQL のかわりに作成され、1 回のみ使用するためにアプリケーションで生成されたアウトラインに DROP\_UNUSED を使用場合があります。そのような文では、アウトラインは使用されず、貴重なディスク領域を単に占有するだけです。

## DROP\_BY\_CAT プロシージャ

このプロシージャは、特定のカテゴリに属しているすべてのアウトラインを削除します。

### 構文

```
OUTLN_PKG.DROP_BY_CAT (  
    cat VARCHAR2);
```

### パラメータ

表 55-2 DROP\_BY\_CAT プロシージャのパラメータ

パラメータ	説明
cat	削除するアウトラインのカテゴリ。

### 使用上の注意

アウトラインのカテゴリで、時々バージが必要になることがあります。このプロシージャは、それを 1 回のコールで実行します。

### 例

この例では、DEFAULT カテゴリ内にあるすべてのアウトラインを削除します。

```
OUTLN_PKG.DROP_BY_CAT('DEFAULT');
```

## UPDATE\_BY\_CAT プロシージャ

このプロシージャは、あるカテゴリにあるすべてのアウトラインのカテゴリを、新規のカテゴリに変更します。アウトライン内の SQL テキストがターゲット・カテゴリ内ですでにアウトラインを持っている場合は、新規カテゴリにマージされません。

### 構文

```
OUTLN_PKG.UPDATE_BY_CAT (  
    oldcat VARCHAR2 DEFAULT 'DEFAULT',  
    newcat VARCHAR2 DEFAULT 'DEFAULT');
```

パラメータ

表 55-3 UPDATE\_BY\_CAT プロシージャのパラメータ

パラメータ	説明
oldcat	変更する現行のカテゴリ。
newcat	アウトラインを変更するターゲット・カテゴリ。

使用上の注意

アウトラインのセットが希望通りならば、アウトラインを実験的なカテゴリから本番カテゴリに移動することが選択できます。同様に、あるカテゴリから別の既存のカテゴリに、アウトラインのセットをマージすることもできます。

例

次の例では、DEFAULT カテゴリ内のすべてのアウトラインを CAT1 カテゴリに変更します。

```
OUTLN_PKG.UPDATE_BY_CAT( 'DEFAULT' , 'CAT1' );
```



UTL\_COLL パッケージによって、問合せや更新を行うためのコレクション・ロケータを PL/SQL プログラムで使用できます。

# サブプログラムの要約

現在、このパッケージでサポートしているファンクションは、IS\_LOCATOR のみです。

## IS\_LOCATOR ファンクション

このファンクションは、コレクション項目が実際にロケータかどうかを判別します。

### 構文

```
UTL_COLL.IS_LOCATOR (  
    collection IN ANY)  
RETURNS BOOLEAN;
```

### パラメータ

表 56-1 IS\_LOCATOR ファンクションのパラメータ

パラメータ	説明
collection	ネストした表または VARRAY 項目。

### 戻り値

表 56-2 IS\_LOCATOR ファンクションの戻り値

戻り値	説明
1	コレクション項目はロケータです。
0	コレクション項目はロケータではありません。

### プラグマ

WNDS、WNPS および RNPS の各プラグマの断言。

### 例外

なし。

## 例

```
CREATE OR REPLACE TYPE list_t as TABLE OF VARCHAR2(20);
/

CREATE OR REPLACE TYPE phone_book_t AS OBJECT (
    pno number,
    ph list_t );
/

CREATE TABLE phone_book OF phone_book_t
    NESTED TABLE ph STORE AS nt_ph;
CREATE TABLE phone_book1 OF phone_book_t
    NESTED TABLE ph STORE AS nt_ph_1 RETURN LOCATOR;

INSERT INTO phone_book VALUES(1, list_t('650-633-5707','650-323-0953'));
INSERT INTO phone_book1 VALUES(1, list_t('415-555-1212'));

CREATE OR REPLACE PROCEDURE chk_coll IS
    plist list_t;
    plist1 list_t;
BEGIN
    SELECT ph INTO plist FROM phone_book WHERE pno=1;

    SELECT ph INTO plist1 FROM phone_book1 WHERE pno=1;

    IF (UTL_COLL.IS_LOCATOR(plist)) THEN
        DBMS_OUTPUT.PUT_LINE('plist is a locator');
    ELSE
        DBMS_OUTPUT.PUT_LINE('plist is not a locator');
    END IF;

    IF (UTL_COLL.IS_LOCATOR(plist1)) THEN
        DBMS_OUTPUT.PUT_LINE('plist1 is a locator');
    ELSE
        DBMS_OUTPUT.PUT_LINE('plist1 is not a locator');
    END IF;

END chk_coll;

SET SERVEROUTPUT ON
EXECUTE chk_coll;
```



UTL\_FILE パッケージによって、ユーザーは PL/SQL プログラムでオペレーティング・システム (OS) のテキスト・ファイルの読取りと書込みができます。このパッケージは、標準 OS のストリーム・ファイル入出力 (I/O) の制限付きバージョンを提供します。

このファイル I/O 機能は、標準オペレーティング・システムのストリーム・ファイル I/O (OPEN、GET、PUT、CLOSE) と同様ですが、いくつかの制限があります。たとえば、FOPEN ファンクションをコールすると、ファイル・ハンドルが戻されます。後続の GET\_LINE または PUT のコールでこのファイル・ハンドルを使用し、ファイルへのストリーム I/O を実行します。ファイルの I/O が終了した場合は、FCLOSE をコールして出力を完了し、そのファイルに関連しているリソースを解放します。

---

## セキュリティ

PL/SQL のファイル I/O 機能は、PL/SQL のクライアント側とサーバー側の両方で使用可能です。クライアント・インプリメンテーション（テキスト I/O）は、オペレーティング・システムによる通常のファイル許可検査の対象であるため、追加セキュリティ制約は不要です。ただし、サーバー・インプリメンテーションは権限モードで稼働する場合があります、この機能の範囲を制限するための追加セキュリティ制約を必要とします。

---

**注意：** UTL\_FILE パッケージは、Oracle Procedure Builder が提供しているクライアント側の TEXT\_IO パッケージと類似しています。サーバー・インプリメンテーションに対する制約事項には、UTL\_FILE と TEXT\_IO の間である程度の API 差異が必要です。PL/SQL のファイル I/O では、PL/SQL 例外を使用して、ユーザーにエラーが戻されます。

---

### サーバーのセキュリティ

PL/SQL のファイル I/O に対するサーバー・セキュリティは、アクセス可能なディレクトリへの制限で構成されています。アクセス可能なディレクトリは、インスタンス・パラメータの初期設定ファイル（INIT.ORA）で指定する必要があります。

次のように UTL\_FILE\_DIR パラメータを使用して、初期設定ファイル内の UTL\_FILE ファンクションにアクセス可能なディレクトリを指定します。

```
UTL_FILE_DIR = <directory name>
```

---

**注意：** ディレクトリの仕様部は、プラットフォームによって異なります。

---

インスタンスの初期設定ファイルに行 `UTL_FILE_DIR = /usr/jsmith/my_app` が含まれている場合、ディレクトリ `/usr/jsmith/my_app` は、FOPEN ファンクションからアクセス可能です。大小文字を区別するオペレーティング・システムでは、`/usr/jsmith/My_App` という名前のディレクトリはアクセス不可になることに注意してください。

パラメータ仕様部の `UTL_FILE_DIR = *` には特別な意味があります。このエントリは、ディレクトリのアクセス検査をオフにして、UTL\_FILE ファンクションからすべてのディレクトリにアクセスできるようにします。

---

---

### 注意：

'\*' オプションの使用には、細心の注意を払ってください。このオプションは、本番のシステムでは使用しないことをお勧めします。また、アクセス可能ディレクトリのリストに '\*' (UNIX の現行ディレクトリ) を含めないでください。

シンボリック・リンクが使用可能なファイル・システム上でセキュリティを保証するために、PL/SQL ファイル I/O ファンクションでアクセスできるディレクトリへの書き込み許可をユーザーに与えないでください。シンボリック・リンクと PL/SQL ファイル I/O は、通常オペレーティング・システムの許可検査を迂回するために使用できます。また、他の方法ではアクセスできないディレクトリにユーザーが読み込みまたは書き込みアクセスをできるようにします。

---

---

## ファイルの所有権と保護

UNIX システムでは、FOPEN ファンクションで作成したファイルにはそのファイルの所有者、つまりインスタンスを実行するシャドウ・プロセスの所有者がいます。通常の場合、この所有者は oracle です。FOPEN で作成したファイルは、UTL\_FILE サブプログラムを使用して常に読み込みと書き込みができますが、非権限ユーザーがこれらのファイルを PL/SQL 以外で読み込む場合は、システム管理者によるアクセス権の設定が必要です。

## 例 (UNIX 固有)

パラメータ初期設定ファイルに次の行のみ含まれている場合は、

```
UTL_FILE_DIR=/appl/gl/log
UTL_FILE_DIR=/appl/gl/out
```

次のファイルの場所とファイル名が有効です。

FILE LOCATION	FILENAME
/appl/gl/log	L10324.log
/appl/gl/out	O10324.out

ただし、次のファイルの場所とファイル名は無効です。

FILE LOCATION	FILENAME	
/appl/gl/log/backup	L10324.log	# subdirectory
/APPL/gl/log	L10324.log	# uppercase
/appl/gl/log	backup/L10324.log	# dir in name
/usr/tmp	T10324.tmp	# not in INIT.ORA

---

---

**注意：** ユーザー・レベルのファイル許可はありません。UTL\_FILE\_DIR パラメータで指定されたすべてのファイルの場所は、ファイル I/O プロシージャのすべてのユーザーが、読み込みと書き込みのために使用できます。これによって、オペレーティング・システムのファイル許可を上書きできます。

---

---

**型**

```
TYPE file_type IS RECORD (id BINARY_INTEGER);
```

FILE\_TYPE の内容は、UTL\_FILE パッケージ専用です。このパッケージのユーザーは、このレコードのコンポーネントの参照または変更は行わないでください。

**例外**

**表 57-1 UTL\_FILE パッケージの例外**

例外名	説明
INVALID_PATH	ファイルの場所またはファイル名が無効です。
INVALID_MODE	FOPEN の open_mode パラメータが無効です。
INVALID_FILEHANDLE	ファイル・ハンドルが無効です。
INVALID_OPERATION	要求どおりにファイルをオープンできないか、または操作できません。
READ_ERROR	読み込み操作中にオペレーティング・システムのエラーが発生しました。
WRITE_ERROR	書き込み操作中にオペレーティング・システムのエラーが発生しました。
INTERNAL_ERROR	PL/SQL 内の未指定エラー。

これらのパッケージ例外に加えて、UTL\_FILE パッケージ内のプロシージャが、NO\_DATA\_FOUND や VALUE\_ERROR などの事前定義済みの PL/SQL 例外を発生させることもあります。



## サブプログラムの要約

表 57-2 UTL\_FILE のサブプログラム

サブプログラム	説明
<a href="#">FOPEN ファンクション</a> (57-6 ページ)	デフォルトの行サイズで入力用または出力用ファイルをオープンします。
<a href="#">IS_OPEN ファンクション</a> (57-7 ページ)	ファイル・ハンドルが、オープンしているファイルを参照しているかどうかを判別します。
<a href="#">FCLOSE プロシージャ</a> (57-8 ページ)	ファイルをクローズします。
<a href="#">FCLOSE_ALL プロシージャ</a> (57-8 ページ)	オープンしているファイル・ハンドルをすべてクローズします。
<a href="#">GET_LINE プロシージャ</a> (57-9 ページ)	オープンしているファイルから、テキストを 1 行読み込みます。
<a href="#">PUT プロシージャ</a> (57-10 ページ)	ファイルに 1 行を書き込みます。行終了記号は追加されません。
<a href="#">NEW_LINE プロシージャ</a> (57-11 ページ)	1 つ以上の OS 固有の行終了記号をファイルに書き込みます。
<a href="#">PUT_LINE プロシージャ</a> (57-12 ページ)	ファイルに 1 行を書き込みます。OS 固有の行終了記号が追加されます。
<a href="#">PUTF プロシージャ</a> (57-12 ページ)	フォーマット付きの PUT プロシージャです。
<a href="#">FFLUSH プロシージャ</a> (57-14 ページ)	保留中のすべての出力データを物理的にファイルへ書き込みます。
<a href="#">FOPEN ファンクション</a> (57-14 ページ)	指定した最大行サイズでファイルをオープンします。

FOPEN ファンクション

このファンクションは、入力用または出力用にファイルをオープンします。ファイルの場所は、インスタンスの初期化パラメータ UTL\_FILE\_DIR で定義されているアクセス可能なディレクトリである必要があります。ディレクトリのパスは事前に存在している必要があります、FOPEN では作成されません。

FOPEN は、ファイル・ハンドルを戻します。このファイル・ハンドルは、ファイル上のすべての後続 I/O 操作で使用する必要があります。

このバージョンの FOPEN は、最大行サイズのパラメータを使用しません。したがって、デフォルト（ほとんどのシステムで 1023）が使用されます。異なる最大行サイズを指定するには、二重定義されたバージョンの「FOPEN ファンクション」( 57-14 ページ ) を使用します。

最大 50 ファイルまで同時にオープンできます。

構文

```
UTL_FILE.FOPEN (
    location  IN VARCHAR2,
    filename  IN VARCHAR2,
    open_mode IN VARCHAR2)
RETURN UTL_FILE.FILE_TYPE;
```

パラメータ

表 57-3 FOPEN ファンクションのパラメータ

パラメータ	説明
location	ファイルをオープンするディレクトリを指定する、オペレーティング・システム固有の文字列。
filename	ファイルの名前。拡張子（ファイル・タイプ）が含まれ、ディレクトリ・パス情報はありません。（UNIX オペレーティング・システムでは、ファイル名を '/' で終了させることはできません）。
open_mode	ファイルのオープン方法を指定する文字列（大文字も小文字も使用できます）。  サポートされている値、およびその値を指定して使用できる UTL_FILE プロシージャは次のとおりです。  'r' テキストの読み込み（GET_LINE）  'w' テキストの書き込み（PUT、PUT_LINE、NEW_LINE、PUTF、FFLUSH）  'a' テキストの追加（PUT、PUT_LINE、NEW_LINE、PUTF、FFLUSH）

---

---

**注意：** `open_mode` の値 'a' を使用して、存在しないファイルをオープンすると、そのファイルは書込み ('w') モードで作成されます。

---

---

## 戻り値

`FOPEN` は、そのファイルを操作する後続プロシージャすべてに渡す必要のあるファイル・ハンドルを戻します。ファイル・ハンドルの特定の内容は、`UTL_FILE` パッケージ専用であり、`UTL_FILE` のユーザーは、個々のコンポーネントの参照または変更を行わないでください。

---

---

**注意：** ファイルの場所とファイル名の各パラメータは、別々の文字列として `FOPEN` ファンクションに指定されるため、初期化ファイルで指定したとおりに、アクセス可能なディレクトリのリストと照合してファイルの場所をチェックできます。ファイルの場所と名前ですystem上の正しいファイル名を示し、そのディレクトリがアクセス可能であることが必要です。アクセス可能なディレクトリのサブディレクトリは、必ずしもアクセス可能である必要はありません。サブディレクトリも初期設定ファイルの完全パス名を使用して指定する必要があります。

UNIX での C シェル環境変数などのオペレーティング・システム固有のパラメータは、ファイルの場所またはファイル名のパラメータでは使用できません。

---

---

## 例外

`INVALID_PATH`  
`INVALID_MODE`  
`INVALID_OPERATION`

## IS\_OPEN ファンクション

このファンクションは、オープンしているファイルをファイル・ハンドルが識別するかどうかをテストします。`IS_OPEN` は、ファイル・ハンドルが、オープンしたがクローズしていないファイルを示すかどうかのみ報告します。このファンクションは、ユーザーがファイル・ハンドルを使用しようとしたときに、オペレーティング・システムのエラーが発生しないことを保証するものではありません。

## 構文

```
UTL_FILE.IS_OPEN (  
    file IN FILE_TYPE)  
RETURN BOOLEAN;
```

パラメータ

表 57-4 IS\_OPEN ファンクションのパラメータ

パラメータ	説明
file	FOPEN コールが戻すアクティブなファイル・ハンドル。

戻り値

TRUE または FALSE。

例外

なし。

FCLOSE プロシージャ

このプロシージャは、ファイル・ハンドルが示すオープン・ファイルをクローズします。  
FCLOSE の実行時に、まだ書き込んでいないデータがバッファに残っていると、ファイルのクローズ時に WRITE\_ERROR 例外を受け取る場合があります。

構文

```
UTL_FILE.FCLOSE (  
    file IN OUT FILE_TYPE);
```

パラメータ

表 57-5 FCLOSE プロシージャのパラメータ

パラメータ	説明
file	FOPEN コールが戻すアクティブなファイル・ハンドル。

例外

```
WRITE_ERROR  
INVALID_FILEHANDLE
```

FCLOSE\_ALL プロシージャ

このプロシージャは、セッションでオープンしているすべてのファイル・ハンドルをクローズします。これは、PL/SQL プログラムの例外終了などの非常時のクリーン・アップ・プロシージャとして使用します。

---

**注意:** FCLOSE\_ALL は、ユーザーが保持しているオープン・ファイル・ハンドルの状態は変更しません。つまり、ファイルはクローズされていても、FCLOSE\_ALL コール後のファイル・ハンドルの IS\_OPEN テストでは TRUE が戻されます。FCLOSE\_ALL の前にオープンされたファイルには、以降の読み込み操作や書き込み操作を行うことができません。

---

## 構文

```
UTL_FILE.FCLOSE_ALL;
```

## パラメータ

なし。

## 例外

```
WRITE_ERROR
```

## GET\_LINE プロシージャ

このプロシージャは、ファイル・ハンドルが示すオープン・ファイルからテキストを 1 行読み込んで、出力バッファ・パラメータに配置します。テキストは、ファイルの終わりまで読み込まれますが、行の終了記号は含まれません。

行がバッファに収まらない場合は、VALUE\_ERROR 例外が発生します。" ファイルの終わり " に到達したためにテキストが読み込まれなかった場合は、NO\_DATA\_FOUND 例外が発生します。

行終了記号の文字はバッファに読み込まれないため、ブランク行を読み込むと空の文字列が戻されます。

FOPEN の二重定義されたバージョンで大きいサイズを指定しない限り、入力レコードの最大サイズは 1023 バイトです。

## 構文

```
UTL_FILE.GET_LINE (  
    file           IN  FILE_TYPE,  
    buffer         OUT VARCHAR2);
```

パラメータ

表 57-6 GET\_LINE プロシージャのパラメータ

パラメータ	説明
file	FOPEN コールが戻すアクティブなファイル・ハンドル。  ファイルは読みみ用（モード 'r'）としてオープンする必要があります。そうでない場合は、INVALID_OPERATION 例外が発生します。
buffer	ファイルから読み込んだ行を受け取るデータ・バッファ。

例外

INVALID\_FILEHANDLE  
INVALID\_OPERATION  
READ\_ERROR  
NO\_DATA\_FOUND  
VALUE\_ERROR

PUT プロシージャ

PUT プロシージャは、ファイル・ハンドルが示すオープン・ファイルに、バッファ・パラメータ内に格納されているテキスト文字列を書き込みます。このファイルは書込み操作用にオープンされる必要があります。PUT は、行終了記号を追加しません。行の終了には NEW\_LINE を使用するか、または PUT\_LINE を使用して行終了記号付きの完全な 1 行を書き込んでください。

二重定義されたバージョンの FOPEN で大きいサイズを指定しない限り、入力レコードの最大サイズは 1023 バイトです。

構文

```
UTL_FILE.PUT (  
    file      IN FILE_TYPE,  
    buffer    IN VARCHAR2);
```

## パラメータ

表 57-7 PUT プロシージャのパラメータ

パラメータ	説明
file	FOPEN コールが戻すアクティブなファイル・ハンドル。
buffer	ファイルに書き込むテキストを含んだバッファ。 ファイルはモード 'w' またはモード 'a' を使用してオープンする必要があります。これ以外のモードを使用すると、INVALID_OPERATION 例外が発生します。

## 例外

INVALID\_FILEHANDLE  
INVALID\_OPERATION  
WRITE\_ERROR

## NEW\_LINE プロシージャ

このプロシージャは、入力ファイル・ハンドルが示すファイルに、1 つ以上の行終了記号を書き込みます。行終了記号はプラットフォーム固有の文字や文字列であるため、このプロシージャは PUT とは異なります。

## 構文

```
UTL_FILE.NEW_LINE (  
    file      IN FILE_TYPE,  
    lines     IN NATURAL := 1);
```

## パラメータ

表 57-8 NEW\_LINE プロシージャのパラメータ

パラメータ	説明
file	FOPEN コールが戻すアクティブなファイル・ハンドル。
lines	ファイルに書き込む行終了記号の数。

## 例外

INVALID\_FILEHANDLE  
INVALID\_OPERATION  
WRITE\_ERROR

## PUT\_LINE プロシージャ

このプロシージャは、ファイル・ハンドルが示すオープン・ファイルに、バッファ・パラメータ内に格納されているテキスト文字列を書き込みます。ファイルは書き込み操作にオープンされる必要があります。PUT\_LINE は、プラットフォーム固有の行終了記号の文字または文字列で行を終了します。

FOPEN の二重定義バージョンで大きい値を指定しない限り、出力レコードの最大サイズは 1023 バイトです。

### 構文

```
UTL_FILE.PUT_LINE (  
    file      IN FILE_TYPE,  
    buffer    IN VARCHAR2);
```

### パラメータ

表 57-9 PUT\_LINE プロシージャのパラメータ

パラメータ	説明
file	FOPEN コールが戻すアクティブなファイル・ハンドル。
buffer	ファイルに書き込む行を含んだテキスト・バッファ。

### 例外

```
INVALID_FILEHANDLE  
INVALID_OPERATION  
WRITE_ERROR
```

## PUTF プロシージャ

このプロシージャは、フォーマットされた PUT プロシージャです。これは、制限付きの printf() のように動作します。フォーマット文字列には任意のテキストを指定できますが、文字 '%s' と '\n' には次のような特別な意味があります。

- %s                   この文字列を引数リスト内の次の引数の文字列値に置き換えます。
- \n                   適切なプラットフォーム固有の行終了記号に置き換えます。



構文

```
UTL_FILE.PUTF (
    file      IN FILE_TYPE,
    format    IN VARCHAR2,
    [arg1     IN VARCHAR2  DEFAULT NULL,
    . . .
    arg5      IN VARCHAR2  DEFAULT NULL]);
```

パラメータ

表 57-10 PUTF プロシージャのパラメータ

パラメータ	説明
file	FOPEN コールが戻すアクティブなファイル・ハンドル。
format	テキストやフォーマット文字 '\n' と '%s' を含むことができるフォーマット文字列。
arg1..arg5	1 ~ 5 個までのオプションの引数文字列。  引数文字列は、フォーマット文字列内の '%s' フォーマットに、順序正しく置き換えられます。  引数より多いフォーマットがフォーマット・パラメータ文字列内にある場合は、引数のない各 '%s' は空の文字列に置き換えられます。

例

次に、行を書き込む例を示します。

```
Hello, world!
I come from Zork with greetings for all earthlings.

my_world varchar2(4) := 'Zork';
...
PUTF(my_handle, 'Hello, world!\nI come from %s with %s.\n',
      my_world,
      'greetings for all earthlings');
```

引数より多い %s フォーマットがフォーマット・パラメータ内にある場合は、対応する引数のない %s は空の文字列に置き換えられます。

例外

```
INVALID_FILEHANDLE
INVALID_OPERATION
WRITE_ERROR
```

## FFLUSH プロシージャ

FFLUSH は、ファイル・ハンドルが示すファイルに、保留中のすべてのデータを物理的に書き込みます。ファイルに書き込むデータは通常バッファリングされます。FFLUSH プロシージャは、バッファリングされているデータを強制的にファイルに書き込みます。

フラッシュは、まだオープンしているファイルを読み込む必要がある場合に役立ちます。たとえば、デバッグ・メッセージをファイルにフラッシュして、即時に読み込むことができます。

### 構文

```
UTL_FILE.FFLUSH (  
    file IN FILE_TYPE);  
invalid_maxlinesize EXCEPTION;
```

### パラメータ

表 57-11 FFLUSH プロシージャのパラメータ

パラメータ	説明
file	FOPEN コールが戻すアクティブなファイル・ハンドル。

### 例外

```
INVALID_FILEHANDLE  
INVALID_OPERATION  
WRITE_ERROR
```

## FOPEN ファンクション

このファンクションは、ファイルをオープンします。最大 50 ファイルまで同時にオープンできます。

**注意：** このバージョンの FOPEN では、最大行サイズを任意に指定できます。別のバージョンの「[FOPEN ファンクション](#)」( 57-6 ページ ) では、デフォルトの行サイズが使用されます。

## 構文

```
UTL_FILE.FOPEN (  
    location      IN VARCHAR2,  
    filename      IN VARCHAR2,  
    open_mode     IN VARCHAR2,  
    max_linesize  IN BINARY_INTEGER)  
RETURN file_type;
```

## パラメータ

表 57-12 FOPEN ファンクションのパラメータ

パラメータ	説明
location	ファイルのディレクトリ位置。
filename	ファイル名（拡張子を含む）。
open_mode	オープン・モード（'r'、'w'、'a'）。
max_linesize	改行文字を含むこのファイルの 1 行あたりの最大文字数（最小値は 1、最大値は 32767）。

## 戻り値

表 57-13 FOPEN ファンクションの戻り値

戻り値	説明
file_type	オープン・ファイルのハンドル。

## 例外

INVALID\_PATH: ファイルの場所またはファイル名が無効です。

INVALID\_MODE: open\_mode の文字列が無効です。

INVALID\_OPERATION: ファイルを要求どおりにオープンできません。

INVALID\_MAXLINESIZE: 指定した max\_linesize が大きすぎるか、または小さすぎます。



UTL\_HTTP は、PL/SQL と SQL からハイパー・テキスト転送プロトコル (HTTP) のコールアウトを行います。ユーザーはこのファンクションを使用して、インターネット上のデータにアクセスしたり、Oracle Web Server カートリッジをコールできます。

UTL\_HTTP には、REQUEST と REQUEST\_PIECES の 2 つの類似したエントリポイントが含まれています。各ファンクションは、ユニバーサル・リソース・ロケータ (URL) の文字列を取得し、サイトに接続して、そのサイトで取得したデータ (一般的には HTML、ハイパー・テキスト・マークアップ言語) を戻します。

# 例外

表 58-1 UTL\_HTTP パッケージの例外

例外	説明
INIT_FAILED	HTTP コールアウト・サブシステムの初期化に失敗しました（使用可能メモリーの不足などの環境的な理由による）。
REQUEST_FAILED	HTTP コールに失敗しました（たとえば、HTTP デーモンに失敗した場合、REQUEST や REQUEST_PIECES への引数が NULL か HTTP 以外の構文であるために、URL として解釈できない場合など）。

前述の 2 つの例外は、例外ハンドラで明示的に取得しない限り、次のメッセージがレポートされます。ORA-06510: PL/SQL: ユーザー定義の例外が処理されませんでした。  
これらの例外はこのシステム・パッケージで定義されますが、ユーザー定義としてレポートされます。

HTTP 要求の処理中にその他の例外状況が発生した場合（メモリー不足エラーなど） フังก์ション REQUEST または REQUEST\_PIECES は、その例外を再発生させます。

指定した URL への要求から応答がない場合（たとえば、URL に対応するサイトに接続できないため）は、フォーマットされた HTML エラー・メッセージが戻されます。次に例を示します。

```
<HTML>
<HEAD>
<TITLE>Error Message</TITLE>
</HEAD>
<BODY>
<H1>Fatal Error 500</H1>
Can't Access Document:  http://home.nothing.comm.
<P>
<B>Reason:</B> Can't locate remote host:  home.nothing.comm.
<P>

<P><HR>
<ADDRESS><A HREF="http://www.w3.org">
CERN-HITPD3.0A</A></ADDRESS>
</BODY>
</HTML>
```

## 使用上の注意

同一マシン（同一の権限や環境変数など）でブラウザを使用して URL に接続できない場合、REQUEST または REQUEST\_PIECES で、その URL への接続が成功することは期待できません。

REQUEST または REQUEST\_PIECES に失敗した場合（たとえば、例外が発生した場合や HTML フォーマットのエラー・メッセージが戻された場合で、URL 引数は正しいと考えられる場合）は、ブラウザで同じ URL に接続を試みて、ユーザーのマシンからネットワークの可用性を検証します。ユーザーのブラウザにプロキシ・サーバー・セットがある場合、そのセットは、オプションの proxy パラメータを使用して REQUEST または REQUEST\_PIECES の各コールごとに設定する必要があります。

**注意：** UTL\_HTTP では、環境変数を使用してそのプロキシ動作を指定することもできます。たとえば、UNIX では、環境変数 http\_proxy を URL に設定することによって、そのサービスを HTTP 要求のプロキシ・サーバーとして使用するよう指定します。また、環境変数 no\_proxy をドメイン名に設定することによって、そのドメインの URL では HTTP プロキシ・サーバーを使用しないように指定します。

## サブプログラムの要約

表 58-2 UTL\_HTTP パッケージのサブプログラム

サブプログラム	説明
<a href="#">REQUEST ファンクション</a> (58-3 ページ)	指定した URL から取り出したデータの最初の 2000 バイトまでを戻します。
<a href="#">REQUEST_PIECES ファンクション</a> (58-5 ページ)	指定した URL から取り出したデータについて、2000 バイト・ピースの PL/SQL 表を戻します。

## REQUEST ファンクション

このファンクションは、指定した URL から取り出したデータの最初の 2000 バイトまでを戻します。

### 構文

```
UTL_HTTP.REQUEST (  
    url      IN VARCHAR2,  
    proxy    IN VARCHAR2 DEFAULT NULL)  
RETURN VARCHAR2;
```

プラグマ

```
pragma restrict_references (request, wnds, rnds, wnps, rmps);
```

パラメータ

表 58-3 REQUEST ファンクションのパラメータ

パラメータ	説明
url	ユニバーサル・リソース・ロケータ。
proxy	( オプション ) HTTP 要求時に使用するプロキシ・サーバーを指定します。

戻り値

戻り型は長さ 2000 以下の文字列で、引数 URL への HTTP 要求から戻された HTML 結果から最初の 2000 バイトまでが含まれます。

例外

```
INIT_FAILED
REQUEST_FAILED
```

例

```
SVRMGR> SELECT utl_http.request('http://www.oracle.com/') FROM dual;
UTL_HTTP.REQUEST('HTTP://WWW.ORACLE.COM/')
<html>
<head><title>Oracle Corporation Home Page</title>
<!--changed Jan. 16, 19
1 row selected.
```

firewall の内側にいる場合は、proxy パラメータを含めます。たとえば、Oracle firewall 内には、www-proxy.us.oracle.com という名前のプロキシ・サーバーがあります。

```
SVRMGR> SELECT
utl_http.request('http://www.oracle.com', 'www-proxy.us.oracle.com') FROM dual;
```



# REQUEST\_PIECES ファンクション

このファンクションは、指定した URL から取り出したデータについて、2000 バイト・ピースの PL/SQL 表を戻します。

## 構文

type html\_pieces is table of varchar2(2000) index by binary\_integer;

```
UTL_HTTP.REQUEST_PIECES (  
    url          IN VARCHAR2,  
    max_pieces   NATURAL      DEFAULT 32767,  
    proxy        IN VARCHAR2  DEFAULT NULL)  
RETURN HTML_PIECES;
```

## プラグマ

pragma restrict\_references (request\_pieces, wnds, rnds, wnps, rnps);

## パラメータ

表 58-4 REQUEST\_PIECES ファンクションのパラメータ

パラメータ	説明
url	ユニバーサル・リソース・ロケータ。
max_pieces	( オプション ) REQUEST_PIECES が戻すピースの最大数 ( 長さは各 2000 文字、最後のピースはこれより短くなる場合があります )。指定する場合、引数は正の整数を指定します。
proxy	( オプション ) HTTP 要求時に使用するプロキシ・サーバーを指定します。

## 戻り値

REQUEST\_PIECES は、UTL\_HTTP.HTML\_PIECES 型の PL/SQL 表を戻します。PL/SQL 表の各要素は、長さ 2000 の文字列です。最後の要素は、2000 文字より短くなる場合があります。

REQUEST\_PIECES で戻される PL/SQL 表の要素は、その URL への HTTP 要求から取得されたデータの連続したピースです。

## 例外

```
INIT_FAILED  
REQUEST_FAILED
```

## 例

REQUEST\_PIECES のコールの例は、次のようになります。戻されるピースの数 (0 以上) を調べるための PL/SQL 表メソッド COUNT の使用方法に注意してください。

```
DECLARE pieces utl_http.html_pieces;
BEGIN
    pieces := utl_http.request_pieces('http://www.oracle.com/');
    FOR i in 1 .. pieces.count loop
        .... -- process each piece
    END LOOP;
END;
```

## 例

次のブロックは、URL から最大 100 ピースまでのデータ (最後のピースを除いて各 2000 バイト) を取り出します。取り出すピースの数と、取り出すデータの合計バイト長を出力します。

```
SET SERVEROUTPUT ON
/
DECLARE
    x utl_http.html_pieces;
BEGIN
    x := utl_http.request_pieces('http://www.oracle.com/', 100);
    dbms_output.put_line(x.count || ' pieces were retrieved.');
```

Output
4 pieces were retrieved.
with total length
7687

```
    dbms_output.put_line('with total length ');
    IF x.count < 1
    THEN dbms_output.put_line('0');
    ELSE dbms_output.put_line
        ((2000 * (x.count - 1)) + length(x(x.count)));
    END IF;
END;
/
-- Output
Statement processed.
4 pieces were retrieved.
with total length
7687
```

UTL\_RAW パッケージは、RAW データ型を操作するための SQL ファンクションを提供します。通常の SQL ファンクションは RAW で作動せず、PL/SQL では RAW データ型と CHAR データ型の間でオーバーロードできないため、このパッケージが必要になります。UTL\_RAW には、各種の COBOL 数値書式を複数の RAW の間で変換するサブプログラムも含まれます。

UTL\_RAW は、データベース環境に固有ではなく、他の環境でもデータベース環境と同じように実際に使用できます。このため、DBMS のかわりに、UTL という接頭辞がパッケージに付けられます。

使用上の注意

RAW ファンクションには、多くの使用方法があります。UTL\_RAW により、RAW レコードは多くの要素で構成できます。RAW データ型を使用すると、キャラクタ・セット変換は実行されず、RAW は、リモート・プロシージャ・コール (RPC) を介して転送されるときに元の書式で保持されます。

また、RAW ファンクションによって、以前は hextoraw ファンクションと rawtohex ファンクションに限定されていたバイナリ・データを操作できます。

サブプログラムの要約

表 59-1 UTL\_RAW パッケージのサブプログラム

サブプログラム	説明
<a href="#">CONCAT ファンクション</a> (59-3 ページ)	最大 12 までの RAW を単一の RAW に連結します。
<a href="#">CAST_TO_RAW ファンクション</a> (59-4 ページ)	n データ・バイトを使用して表した VARCHAR2 を、n データ・バイトを持つ RAW に変換します。
<a href="#">CAST_TO_VARCHAR2 ファンクション</a> (59-5 ページ)	n データ・バイトを使用して表した RAW を、n データ・バイトを持つ VARCHAR2 に変換します。
<a href="#">LENGTH ファンクション</a> (59-6 ページ)	RAW r の長さをバイトで戻します。
<a href="#">SUBSTR ファンクション</a> (59-7 ページ)	pos から開始して、RAW r から len バイトを戻します。
<a href="#">TRANSLATE ファンクション</a> (59-8 ページ)	RAW from_set と to_set の変換によるバイトに従って、入力 RAW r 内のバイトを変換します。
<a href="#">TRANSLITERATE ファンクション</a> (59-10 ページ)	RAW from_set と to_set の文字変換によるバイトに従って、入力 RAW r 内のバイトを変換します。
<a href="#">OVERLAY ファンクション</a> (59-11 ページ)	ターゲット RAW の指定部分を overlay RAW でオーバーレイし、ターゲットのバイト位置 pos から始まる len バイト分を処理します。
<a href="#">COPIES ファンクション</a> (59-13 ページ)	r を n 回コピーして連結したものを戻します。
<a href="#">XRANGE ファンクション</a> (59-14 ページ)	値 start_byte で始まり値 end_byte で終わる、連続した有効な 1 バイト・コードをすべて含む RAW を戻します。
<a href="#">REVERSE ファンクション</a> (59-15 ページ)	RAW r のバイトの順序を、最後から最初に逆転させます。

表 59-1 UTL\_RAW パッケージのサブプログラム

サブプログラム	説明
<a href="#">COMPARE</a> ファンクション (59-16 ページ)	RAW <i>r1</i> と RAW <i>r2</i> を比較します。
<a href="#">CONVERT</a> ファンクション (59-17 ページ)	RAW <i>r</i> をキャラクタ・セット <i>from_charset</i> からキャラクタ・セット <i>to_charset</i> に変換し、結果の RAW を戻します。
<a href="#">BIT_AND</a> ファンクション (59-19 ページ)	RAW <i>r1</i> と RAW <i>r2</i> の値でビット単位の論理演算 "AND" を実行し、"AND" 演算後の結果 RAW を戻します。
<a href="#">BIT_OR</a> ファンクション (59-20 ページ)	RAW <i>r1</i> と RAW <i>r2</i> の値でビット単位の論理演算 "OR" を実行し、"OR" 演算後の結果 RAW を戻します。
<a href="#">BIT_XOR</a> ファンクション (59-21 ページ)	RAW <i>r1</i> と RAW <i>r2</i> の値でビット単位の論理演算 "排他 OR" を実行し、"排他 OR" 演算後の結果 RAW を戻します。
<a href="#">BIT_COMPLEMENT</a> ファンクション (59-22 ページ)	RAW <i>r</i> の値でビット単位の論理演算 "補数" を実行し、"補数" 演算後の結果 RAW を戻します。

CONCAT ファンクション

このファンクションは、最大 12 までの RAW を単一の RAW に連結します。連結したサイズが 32K を超える場合は、エラーが戻ります。

構文

```
UTL_RAW.CONCAT (  
    r1  IN RAW DEFAULT NULL,  
    r2  IN RAW DEFAULT NULL,  
    r3  IN RAW DEFAULT NULL,  
    r4  IN RAW DEFAULT NULL,  
    r5  IN RAW DEFAULT NULL,  
    r6  IN RAW DEFAULT NULL,  
    r7  IN RAW DEFAULT NULL,  
    r8  IN RAW DEFAULT NULL,  
    r9  IN RAW DEFAULT NULL,  
    r10 IN RAW DEFAULT NULL,  
    r11 IN RAW DEFAULT NULL,  
    r12 IN RAW DEFAULT NULL)  
RETURN RAW;
```

プラグマ

```
pragma restrict_references(concat, WNDS, RNDS, WNPS, RNPS);
```

パラメータ

r1....r12 は、連結する RAW 項目です。

戻り値

表 59-2 CONCAT ファンクションの戻り値

戻り値	説明
RAW	連結された項目。

エラー

入力値の合計の長さが RAW の最大許容長である 32767 バイトを超えると、エラーが発生します。

CAST\_TO\_RAW ファンクション

このファンクションは、n データ・バイトを使用して表した VARCHAR2 を、n データ・バイトの RAW に変換します。データは変更されませんが、データ型のみ RAW データ型に変換されます。

構文

```
UTL_RAW.CAST_TO_RAW (  
    c IN VARCHAR2)  
    RETURN RAW;
```

プラグマ

```
pragma restrict_references(cast_to_raw, WNDS, RNDS, WNPS, RNPS);
```

パラメータ

表 59-3 CAST\_TO\_RAW ファンクションのパラメータ

パラメータ	説明
c	RAW に変換される VARCHAR2。

## 戻り値

表 59-4 CAST\_TO\_RAW ファンクションの戻り値

戻り値	説明
RAW	先行する長さのフィールドのない、入力 VARCHAR2 と同じバイト長の同じデータ。
NULL	入力パラメータ c が NULL の場合。

## エラー

なし。

## CAST\_TO\_VARCHAR2 ファンクション

このファンクションは、n データ・バイトを使用して表した RAW を、n データ・バイトの VARCHAR2 に変換します。

**注意：** VARCHAR2 への変換時、その VARCHAR2 内の文字に対して現行の NLS キャラクタ・セットが使用されます。

## 構文

```
UTL_RAW.CAST_TO_VARCHAR2 (  
    r IN RAW)  
RETURN VARCHAR2;
```

## プラグマ

```
pragma restrict_references(cast_to_varchar2, WNDS, RNDS, WNPS, RNPS);
```

## パラメータ

表 59-5 CAST\_TO\_VARCHAR2 ファンクションのパラメータ

パラメータ	説明
r	VARCHAR2 に変更する RAW (先行する長さのフィールドなし)。

戻り値

表 59-6 CAST\_TO\_VARCHAR2 ファンクションの戻り値

戻り値	説明
VARCHAR2	入力 RAW と同じデータ。
NULL	入力パラメータ <i>r</i> が NULL の場合。

エラー  
なし。

LENGTH ファンクション

このファンクションは、RAW *r* の長さをバイトで戻します。

構文

```
UTL_RAW.LENGTH (  
    r IN RAW)  
RETURN NUMBER;
```

プラグマ

```
pragma restrict_references(length, WNDS, RNDS, WNPS, RNPS);
```

パラメータ

表 59-7 LENGTH ファンクションのパラメータ

パラメータ	説明
<i>r</i>	長さを測定する RAW バイト・ストリーム。

戻り値

表 59-8 LENGTH ファンクションの戻り値

戻り値	説明
NUMBER	RAW の現行の長さと同じ長さ。

エラー  
なし。



# SUBSTR ファンクション

このファンクションは、RAW *r* から *pos* で始まる *len* バイトを戻します。

## 構文

```
UTL_RAW.SUBSTR (  
    r      IN RAW,  
    pos    IN BINARY_INTEGER,  
    len    IN BINARY_INTEGER DEFAULT NULL)  
RETURN RAW;
```

## プラグマ

```
pragma restrict_references(substr, WNDS, RNDS, WNPS, RNPS);
```

## パラメータ

*pos* が正の値の場合、SUBSTR は *r* の初めからカウントして最初のバイトを検索します。  
*pos* が負の値の場合、SUBSTR は *r* の最後から逆方向にカウントします。値 *pos* は 0 に指定できません。

*len* を省略すると、SUBSTR は *r* の最後までバイトをすべて戻します。値 *len* は 1 未満に指定できません。

表 59-9 SUBSTR ファンクションのパラメータ

パラメータ	説明
<i>r</i>	一部分を抽出する RAW バイト列。
<i>pos</i>	<i>r</i> 内で抽出を開始するバイト位置。
<i>len</i>	<i>r</i> から抽出する、 <i>pos</i> からのバイト数 (オプション)。

## デフォルトとオプション・パラメータ

表 59-10 SUBSTR ファンクションの例外

オプション・パラメータ	説明
<i>len</i>	位置 <i>pos</i> から <i>r</i> の終わりまでの長さ。

戻り値

表 59-11 SUBSTR ファンクションの戻り値

戻り値	説明
portion of r	pos から始まる len バイト長。
NULL	入力パラメータ r が NULL の場合。

エラー

表 59-12 SUBSTR ファンクションのエラー

エラー	説明
VALUE_ERROR	pos = 0 または len < 0 のいずれかです。

TRANSLATE ファンクション

このファンクションは、RAWfrom\_set と to\_set の変換によるバイトに従って、入力 RAW r 内のバイトを変換します。r 内のバイトが from\_set 内のバイトと一致すると、to\_set 内の対応する位置にあるバイトに置換され、一致しないと削除されます。

r 内のバイトが from\_set で未定義の場合は、結果にコピーされます。from\_set にある最初（最左端）のバイトのみ使用されます。後続の複製部分はスキャンされずに無視されます。to\_set が from\_set より短い場合は、from\_set の余分なバイトに対応する変換はなく、r 内に一致するバイトはありません。

**注意：** TRANSLITERATE とは、次の点で異なります。

- 変換 RAW にデフォルトはありません。
- to\_set の変換 RAW で未定義の r バイトは削除されます。
- 結果 RAW は、入力 RAW r より短い場合があります。

構文

```
UTL_RAW.TRANSLATE (  
    r          IN RAW,  
    from_set  IN RAW,  
    to_set    IN RAW)  
RETURN RAW;
```

## プラグマ

```
pragma restrict_references(translate, WND$S, RND$S, WNPS, RNPS);
```

## パラメータ

表 59-13 TRANSLATE ファンクションのパラメータ

パラメータ	説明
r	変換する RAW ソース・バイト列。
from_set	変換する RAW バイト・コード (r にある場合)。
to_set	対応する from_str バイトが変換される RAW バイト・コード。

## 戻り値

表 59-14 TRANSLATE ファンクションの戻り値

戻り値	説明
RAW	変換されたバイト列。

## エラー

表 59-15 TRANSLATE ファンクションのエラー

エラー	説明
VALUE_ERROR	次のいずれかです。 - r が NULL または長さ 0 (あるいはその両方) です。 - from_set が NULL または長さ 0 (あるいはその両方) です。 - to_set が NULL または長さ 0 (あるいはその両方) です。

TRANSLITERATE ファンクション

このファンクションは、RAW from\_set と to\_set の文字変換によるバイトに従って、入力 RAW r 内のバイトを変換します。r 内の連続するバイトが from\_set 内で検索され、見つからない場合は、変更しないまま結果 RAW にコピーされます。見つかった場合、そのバイトは、to\_set の対応するバイト、対応するバイトが存在しない場合は pad バイトのいずれかに結果 RAW 内で置換されます。

r 内のバイトが from\_set で未定義の場合は、結果にコピーされます。from\_set にある最初（最左端）のバイトのみ使用されます。後続の複製部分はスキャンされずに無視されます。結果 RAW は、常に r と同じ長さになります。

to\_set が from\_set より短い場合、選択した from\_set バイトに対応する to\_set バイトがないと、pad バイトが結果 RAW に埋め込まれます（pad バイトを使用して、to\_set が from\_set と同じ長さまで拡張されたようになります）。

**注意：** TRANSLATE とは、次の点で異なります。

- to\_set で未定義の r バイトが埋め込まれます。
- 結果 RAW は、常に入力 RAW r と同じ長さになります。

構文

```
UTL_RAW.TRANSLITERATE (  
    r          IN RAW,  
    to_set     IN RAW DEFAULT NULL,  
    from_set   IN RAW DEFAULT NULL,  
    pad        IN RAW DEFAULT NULL)  
RETURN RAW;
```

プラグマ

```
pragma restrict_references(transliterate, WNDS, RNDS, WNPS, RNPS);
```

パラメータ

表 59-16 TRANSLITERATE ファンクションのパラメータ

パラメータ	説明
r	変換する RAW 入力バイト列。
from_set	変換する RAW バイト・コード（r にある場合）（任意の長さ）。
to_set	対応する from_set バイトが変換される RAW バイト・コード（任意の長さ）。

表 59-16 TRANSLITERATE ファンクションのパラメータ

パラメータ	説明
pad	to-set が from_set より短い場合に使用する 1 バイト。

## デフォルトとオプション・パラメータ

表 59-17 TRANSLITERATE ファンクションのオプション・パラメータ

オプション・パラメータ	説明
from_set	x'00 ~ x'ff.
to_set	NULL 文字列まで。実際は、必要に応じて pad を使用して、from_set の長さまで拡張されます。
pad	x'00'。

## 戻り値

表 59-18 TRANSLITERATE ファンクションの戻り値

戻り値	説明
RAW	変換されたバイト列。

## エラー

表 59-19 TRANSLITERATE ファンクションのエラー

エラー	説明
VALUE_ERROR	R が NULL または長さ 0 (あるいはその両方) です。

## OVERLAY ファンクション

このファンクションは、ターゲット RAW の指定部分をオーバーレイ RAW でオーバーレイし、ターゲットのバイト位置 pos から始まる len バイト分を処理します。

overlay が len バイト未満の場合は、pad バイトを使用して len バイトまで拡張されます。overlay が len バイトを超える場合は、オーバーレイの余分なバイトは無視されます。ターゲットの位置 pos から始まる len バイトがターゲットの長さをを超える場合、ターゲットは拡張されて overlay 全体の長さになります。

len が指定されている場合は、0 以上である必要があります。pos が指定されている場合は、1 以上である必要があります。pos がターゲットの長さを超えている場合、ターゲット

は pad バイトを使用して位置 pos まで埋め込まれ、さらにターゲットは overlay バイトを使用して拡張されます。

構文

```
UTL_RAW.OVERLAY (  
    overlay_str IN RAW,  
    target      IN RAW,  
    pos         IN BINARY_INTEGER DEFAULT 1,  
    len         IN BINARY_INTEGER DEFAULT NULL,  
    pad         IN RAW              DEFAULT NULL)  
RETURN RAW;
```

プラグマ

```
pragma restrict_references(overlay, WNDS, RNDS, WNPS, RNPS);
```

パラメータ

表 59-20 OVERLAY ファンクションのパラメータ

パラメータ	説明
overlay_str	ターゲットをオーバーレイするために使用するバイト列。
target	オーバーレイするバイト列。
pos	オーバーレイを開始する、ターゲット内での位置（1 から番号付けされている）。
len	オーバーレイするターゲット・バイトの数。
pad	オーバーレイ len がオーバーレイ長を超えた場合、または pos がターゲットの長さを超えた場合に使用するパッド・バイト。

デフォルトとオプション・パラメータ

表 59-21 OVERLAY ファンクションのオプション・パラメータ

オプション・パラメータ	説明
pos	1
len	オーバーレイの長さまで。
pad	x'00'

## 戻り値

**表 59-22 OVERLAY ファンクションの戻り値**

戻り値	説明
RAW	指定したとおりにオーバーレイされたターゲットの <code>byte_string</code> 。

## エラー

**表 59-23 OVERLAY ファンクションのエラー**

エラー	説明
VALUE_ERROR	次のいずれかです。 <ul style="list-style-type: none"><li>- オーバーレイが <code>NULL</code> または長さ 0 (あるいはその両方) です。</li><li>- ターゲットが不明か、未定義です。</li><li>- ターゲットの長さが、RAW の最大長を超えました。</li><li>- <code>len &lt; 0</code></li><li>- <code>pos &lt; 1</code></li></ul>

## COPIES ファンクション

このファンクションは、`r` を `n` 回コピーして連結したものを戻します。

### 構文

```
UTL_RAW.COPIES (  
    r IN RAW,  
    n IN NUMBER)  
RETURN RAW;
```

### プラグマ

```
pragma restrict_references(copies, WNDS, RNDS, WNPS, RNPS);
```

## パラメータ

**表 59-24 COPIES ファンクションのパラメータ**

パラメータ	説明
<code>r</code>	コピーする RAW。

表 59-24 COPIES ファンクションのパラメータ

パラメータ	説明
n	RAW をコピーする回数 (必ず正の数で指定)。

戻り値

このファンクションは、n 回コピーした RAW を戻します。

エラー

表 59-25 COPIES ファンクションのエラー

エラー	説明
VALUE_ERROR	次のいずれかです。 - r が不明、NULL または長さ 0 (あるいはその両方) です。 - n < 1 - 結果の長さが、RAW の最大長を超えました。

XRANGE ファンクション

このファンクションは、連続した有効な 1 バイト・コードをすべて含む RAW を戻し、値 start\_byte で始まり値 end\_byte で終わります。start\_byte が end\_byte より大きい場合、結果バイトの連続は start\_byte で始まり、'FF'x から '00'x に折り返して end\_byte で終わります。start\_byte と end\_byte を指定する場合は、単一バイトの RAW である必要があります。

構文

```
UTL_RAW.XRANGE (  
  start_byte IN RAW DEFAULT NULL,  
  end_byte   IN RAW DEFAULT NULL)  
RETURN RAW;
```

プラグマ

```
pragma restrict_references(xrange, WNDS, RNDS, WNPS, RNPS);
```



## パラメータ

表 59-26 XRANGE ファンクションのパラメータ

パラメータ	説明
start_byte	戻される連続値の最初のバイト・コード値。
end_byte	戻される連続値の最後のバイト・コード値。

## デフォルトとオプション・パラメータ

```
start_byte - x'00'
start_byte - x'00'
end_byte   - x'FF'
```

## 戻り値

表 59-27 XRANGE ファンクションの戻り値

戻り値	説明
RAW	連続した有効な 1 バイトの 16 進数コード。

## エラー

なし。

## REVERSE ファンクション

このファンクションは、RAW r のバイトの順序を、最後から最初に逆転させます。たとえば、x'0102F3' は x'F30201' に逆転し、'xyz' は 'zyx' に逆転します。結果の長さは、入力 RAW の長さと同じです。

## 構文

```
UTL_RAW.REVERSE (  
    r IN RAW)  
RETURN RAW;
```

## プラグマ

```
pragma restrict_references(reverse, WNDS, RNDS, WNPS, RNPS);
```

パラメータ

表 59-28 REVERSE ファンクションのパラメータ

パラメータ	説明
r	逆転する RAW。

戻り値

表 59-29 REVERSE ファンクションの戻り値

戻り値	説明
RAW	r の " 逆転 " した値。

エラー

表 59-30 REVERSE ファンクションのエラー

エラー	説明
VALUE_ERROR	R が NULL または長さ 0 (あるいはその両方) です。

COMPARE ファンクション

このファンクションは、RAW r1 と RAW r2 を比較します。r1 と r2 の長さが異なる場合、短い方の RAW は、必要に応じて pad バイトを使用して右側に拡張されます。

構文

```
UTL_RAW.COMPARE (  
    r1  IN RAW,  
    r2  IN RAW,  
    pad IN RAW DEFAULT NULL)  
RETURN NUMBER;
```

プラグマ

```
pragma restrict_references(compare, WNDS, RNDS, WNPS, RNPS);
```

パラメータ

表 59-31 COMPARE ファンクションのパラメータ

パラメータ	説明
r1	比較する 1 番目の RAW で、NULL または長さ 0 (あるいはその両方) が可能。
r2	比較する 2 番目の RAW で、NULL または長さ 0 (あるいはその両方) が可能。
pad	r1 または r2 の短い方を拡張するためのバイト。

デフォルトとオプション・パラメータ

pad - x'00'

戻り値

表 59-32 COMPARE ファンクションの戻り値

戻り値	説明
NUMBER	RAW バイト列が両方とも NULL または等しい場合は、0。または、最初に不一致になったバイト位置 (1 から番号付けされている) と等しい番号。

エラー

なし。

CONVERT ファンクション

このファンクションは、RAW r をキャラクタ・セット from\_charset からキャラクタ・セット to\_charset に変換し、結果の RAW を戻します。

from\_charset と to\_charset は両方とも、Oracle Server に定義されているサポート・キャラクタのセットである必要があります。

構文

```
UTL_RAW.CONVERT (  
    r                IN RAW,  
    to_charset       IN VARCHAR2,  
    from_charset     IN VARCHAR2)  
RETURN RAW;
```

プラグマ

```
pragma restrict_references(convert, WNDS, RNDS, WNPS, RNPS);
```

パラメータ

表 59-33 CONVERT ファンクションのパラメータ

パラメータ	説明
r	変換する RAW バイト列。
to_charset	r が変換される NLS キャラクタ・セットの名前。
from_charset	r が提供される NLS キャラクタ・セットの名前。

戻り値

表 59-34 CONVERT ファンクションの戻り値

戻り値	説明
RAW	指定したキャラクタ・セットに従って変換されたバイト列 r。

エラー

表 59-35 CONVERT ファンクションのエラー

エラー	説明
VALUE_ERROR	次のいずれかです。  - r が不明、NULL または長さ 0 (あるいはその両方) です。  - from_charset または to_charset が不明、NULL または長さ 0 (あるいはその両方) です。  - from_charset または to_charset の名前が無効か、またはサポートされていません。

## BIT\_AND ファンクション

このファンクションは、RAW *r1* と RAW *r2* の値でビット単位の論理演算 "AND" を実行し、"AND" 演算後の結果 RAW を戻します。

*r1* と *r2* の長さが異なる場合、"AND" 演算は、2 つの RAW の短い方の最後のバイトが演算された後に終了し、長い方の RAW で処理されなかった部分は結果の一部として追加されます。したがって、結果の長さは、2 つの入力 RAW の長い方と同じ長さになります。

### 構文

```
UTL_RAW.BIT_AND (  
    r1 IN RAW,  
    r2 IN RAW)  
RETURN RAW;
```

### プラグマ

```
pragma restrict_references(bit_and, WNDS, RNDS, WNPS, RNPS);
```

### パラメータ

表 59-36 BIT\_AND ファンクションのパラメータ

パラメータ	説明
<i>r1</i>	<i>r2</i> と "AND" 演算をする RAW。
<i>r2</i>	<i>r1</i> と "AND" 演算をする RAW。

### 戻り値

表 59-37 BIT\_AND ファンクションの戻り値

戻り値	説明
RAW	<i>r1</i> と <i>r2</i> の "AND" 演算の結果。
NULL	入力パラメータ <i>r1</i> または <i>r2</i> が NULL の場合。

### エラー

なし。

## BIT\_OR ファンクション

このファンクションは、RAW r1 と RAW r2 の値でビット単位の論理演算 "OR" を実行し、"OR" 演算後の結果 RAW を戻します。

r1 と r2 の長さが異なる場合、"OR" 演算は、2 つの RAW の短い方の最後のバイトが演算された後に終了し、長い方の RAW で処理されなかった部分は結果の一部として追加されます。したがって、結果の長さは、2 つの入力 RAW の長い方と同じ長さになります。

### 構文

```
UTL_RAW.BIT_OR (  
    r1 IN RAW,  
    r2 IN RAW)  
RETURN RAW;
```

### プラグマ

```
pragma restrict_references(bit_or, WNDS, RNDS, WNPS, RNPS);
```

### パラメータ

表 59-38 BIT\_OR ファンクションのパラメータ

パラメータ	説明
r1	r2 と "OR" 演算をする RAW。
r2	r1 と "OR" 演算をする RAW。

### 戻り値

表 59-39 BIT\_OR ファンクションの戻り値

戻り値	説明
RAW	r1 と r2 の "OR" 演算の結果。
NULL	入力パラメータ r1 または r2 が NULL の場合。

### エラー

なし。

## BIT\_XOR ファンクション

このファンクションは、RAW *r1* と RAW *r2* の値でビット単位の論理演算 "排他 OR" を実行し、"排他 OR" 演算後の結果 RAW を戻します。

*r1* と *r2* の長さが異なる場合、"排他 OR" 演算は、2 つの RAW の短い方の最後のバイトが演算された後に終了し、長い方の RAW で処理されなかった部分は結果の一部として追加されます。したがって、結果の長さは、2 つの入力 RAW の長い方と同じ長さになります。

### 構文

```
UTL_RAW.BIT_XOR (  
    r1 IN RAW,  
    r2 IN RAW)  
RETURN RAW;
```

### プラグマ

```
pragma restrict_references(bit_xor, WNDS, RNDS, WNPS, RNPS);
```

### パラメータ

表 59-40 BIT\_XOR ファンクションのパラメータ

パラメータ	説明
<i>r1</i>	<i>r2</i> と "排他 OR" 演算をする RAW。
<i>r2</i>	<i>r1</i> と "排他 OR" 演算をする RAW。

### 戻り値

表 59-41 BIT\_XOR ファンクションの戻り値

戻り値	説明
RAW	<i>r1</i> と <i>r2</i> の "排他 OR" 演算の結果。
NULL	入力パラメータ <i>r1</i> または <i>r2</i> が NULL の場合。

### エラー

なし。

BIT\_COMPLEMENT ファンクション

このファンクションは、RAW *r* の値でビット単位の論理演算 " 補数 " を実行し、" 補数 " 演算後の結果 RAW を戻します。結果の長さは、入力 RAW *r* の長さと等しくなります。

構文

```
UTL_RAW.BIT_COMPLEMENT (  
    r IN RAW)  
    RETURN RAW;
```

プラグマ

```
pragma restrict_references(bit_complement, WNDS, RNDS, WNPS, RNPS);
```

パラメータ

表 59-42 BIT\_COMPLEMENT ファンクションのパラメータ

パラメータ	説明
<i>r</i>	" 補数 " 演算を実行する RAW。

戻り値

表 59-43 BIT\_COMPLEMENT ファンクションの戻り値

戻り値	説明
RAW	<i>r1</i> の " 補数 "。
NULL	入力パラメータ <i>r</i> が NULL の場合。

エラー

なし。



Oracle8i は、ユーザー定義の複合型またはオブジェクト型をサポートします。オブジェクト型のインスタンスをオブジェクトと呼びます。オブジェクト型は、列の型としてまたは表の型として使用できます。

オブジェクト表では、表の各行にオブジェクトが格納されています。オブジェクト表にあるオブジェクトは、オブジェクト識別子で一意に識別できます。

参照はオブジェクトへの持続ポインタで、各参照にはオブジェクト識別子を含めることができます。参照は、オブジェクト型の属性にしたり、表の列に格納することができます。参照を指定して、オブジェクトを取り出すことができます。

UTL\_REF パッケージは、参照ベースの操作をサポートするための PL/SQL プロシージャを提供します。SQL と異なり、UTL\_REF プロシージャでは、オブジェクト表名が不明でも汎用型メソッドを書き込むことができます。

---

## 要件

このパッケージを使用するには、プロシージャ・オプションが必要です。このパッケージは、SYS 権限 (connect internal) で作成する必要があります。このパッケージが提供する操作は、パッケージ所有者 SYS ではなく、現在のコール・ユーザーの下で実行されます。

## データ型

オブジェクト型は、ユーザーが定義するか、またはライブラリ型として提供された複合データ型です。次の構文を使用して、オブジェクト型 `employee_type` を作成できます。

```
CREATE TYPE employee_type AS OBJECT (  
    name    VARCHAR2(20),  
    id      NUMBER,  
  
    member function GET_ID  
        (name VARCHAR2)  
        RETURN MEMBER);
```

オブジェクト型 `employee_type` はユーザー定義型で、`name` と `id` の 2 つの属性、およびメンバー・ファンクションの `GET_ID()` が含まれています。

次の SQL 構文を使用して、オブジェクト表を作成できます。

```
CREATE TABLE employee_table OF employee_type;
```

## 例外

UTL\_REF ファンクションの実行中に、さまざまな理由で例外が戻る場合があります。たとえば、次の使用例では例外が発生します。

- 選択したオブジェクトが存在しません。これには、次のいずれかの理由が考えられます。
  1. オブジェクトが削除されたか、または指定した参照が無効です。
  2. オブジェクト表が削除されたか、または存在しません。
- 直列可能トランザクションで、オブジェクトを変更またはロックできません。オブジェクトは、直列可能トランザクションの開始後に、別のトランザクションで変更されました。
- オブジェクトを選択または変更する権限がありません。UTL\_REF サブプログラムのコール側は、選択または変更するオブジェクトに対する適切な権限が必要です。

表 60-1 UTL\_REF の例外

例外	説明
errnum == 942	権限が不十分です。
errnum == 1031	権限が不十分です。
errnum == 8177	直列可能トランザクションの場合は、直列化できません。
errnum == 60	デッドロックが検出されました。
errnum == 1403	データが見つかりません (REF が NULL である場合など)。

UTL\_REF パッケージは、名前の付いた例外を定義しません。特定の例外を捕捉して適切に処理するために、ブロックを処理する例外を定義できます。

## セキュリティ

UTL\_REF パッケージは、サーバー上のストアード PL/SQL プロシージャまたはパッケージとクライアント側の PL/SQL コードからも同様に使用できます。

サーバー上の PL/SQL プロシージャまたはパッケージから起動した場合、UTL\_REF は、REF が指すオブジェクトに対する適切なアクセス権限が実行者にあることを検証します。

**注意：** これは、定義者の権限で操作する、サーバー上の PL/SQL パッケージまたはプロシージャとは対照的で、パッケージ所有者には、必要な操作を実行するための適切な権限が必要です。

したがって、UTL\_REF がユーザーの SYS 権限で定義された場合、ユーザー A が参照からオブジェクトを選択するために UTL\_REF.SELECT を起動すると、ユーザー A (実行者) は、権限をチェックする必要があります。

UTL\_REF は、クライアント側 PL/SQL コードから起動すると、PL/SQL が実行されているクライアント・セッションの権限で操作が行われます。

# サブプログラムの要約

表 60-2 UTL\_REF サブプログラム

サブプログラム	説明
<a href="#">SELECT_OBJECT プロシージャ</a> (60-4 ページ)	参照を指定してオブジェクトを選択します。
<a href="#">LOCK_OBJECT プロシージャ</a> (60-5 ページ)	参照を指定してオブジェクトをロックします。
<a href="#">UPDATE_OBJECT プロシージャ</a> (60-6 ページ)	参照を指定してオブジェクトを更新します。
<a href="#">DELETE_OBJECT プロシージャ</a> (60-7 ページ)	参照を指定してオブジェクトを削除します。

## SELECT\_OBJECT プロシージャ

このプロシージャは、参照を指定してオブジェクトを選択します。選択されたオブジェクトはデータベースから取り出され、その値は PL/SQL の変数 'object' に入れます。このサブプログラムの意味は、次の SQL 文に似ています。

```
SELECT VALUE(t)
INTO object
FROM object_table t
WHERE REF(t) = reference;
```

前述の SQL 文と異なり、このサブプログラムでは、オブジェクトが常駐しているオブジェクト表名を指定する必要はありません。

### 構文

```
UTL_REF.SELECT_OBJECT (
    reference IN REF "<typename>",
    object    IN OUT "<typename>");
```

### パラメータ

表 60-3 SELECT\_OBJECT プロシージャのパラメータ

パラメータ	説明
reference	選択または取り出すオブジェクトへの参照。
object	選択したオブジェクトを格納する PL/SQL 変数。この変数は、参照されたオブジェクトと同じオブジェクト型である必要があります。

戻り値

なし。

プラグマ

なし。

例外

発生する可能性があります。

LOCK\_OBJECT プロシージャ

このプロシージャは、参照を指定してオブジェクトをロックします。さらに、このプロシージャでは、プログラムによって、ロックされたオブジェクトを選択できます。このサブプログラムの意味は、次の SQL 文に似ています。

```
SELECT VALUE(t)
  INTO object
  FROM object_table t
 WHERE REF(t) = reference
  FOR UPDATE;
```

前述の SQL 文と異なり、このサブプログラムでは、オブジェクトが常駐しているオブジェクト表名を指定する必要はありません。オブジェクトの更新または削除前に、オブジェクトをロックする必要はありません。

構文

```
UTL_REF.LOCK_OBJECT (
    reference IN REF "<typename>");

UTL_REF.LOCK_OBJECT (
    reference IN REF "<typename>",
    object    IN OUT "<typename>");
```

パラメータ

表 60-4 LOCK\_OBJECT プロシージャのパラメータ

パラメータ	説明
reference	ロックするオブジェクトの参照。
object	ロックされたオブジェクトを格納する PL/SQL 変数。この変数は、ロックされたオブジェクトと同じオブジェクト型である必要があります。

**戻り値**

なし。

**プラグマ**

なし。

**例外**

発生する可能性があります。

**UPDATE\_OBJECT プロシージャ**

このプロシージャは、参照を指定してオブジェクトを更新します。参照されたオブジェクトは、PL/SQL 変数 'object' に含まれている値で更新されます。このサブプログラムの意味は、次の SQL 文に似ています。

```
UPDATE object_table t
SET VALUE(t) = object
WHERE REF(t) = reference;
```

前述の SQL 文と異なり、このサブプログラムでは、オブジェクトが常駐しているオブジェクト表名を指定する必要はありません。

**構文**

```
UTL_REF.UPDATE_OBJECT (
    reference IN REF "<typename>",
    object    IN    "<typename>");
```

**パラメータ**

**表 60-5 UPDATE\_OBJECT プロシージャのパラメータ**

パラメータ	説明
reference	更新するオブジェクトの参照。
object	オブジェクトの新規の値を含める PL/SQL 変数。この変数は、更新されたオブジェクトと同じオブジェクト型である必要があります。

**戻り値**

なし。

プラグマ

なし。

例外

発生する可能性があります。

DELETE\_OBJECT プロシージャ

このプロシージャは、参照を指定してオブジェクトを削除します。このサブプログラムの意味は、次の SQL 文に似ています。

```
DELETE FROM object_table
WHERE REF(t) = reference;
```

前述の SQL 文と異なり、このサブプログラムでは、オブジェクトが常駐しているオブジェクト表名を指定する必要はありません。

構文

```
UTL_REF.DELETE_OBJECT (
    reference IN REF "<typename>");
```

パラメータ

表 60-6 DELETE\_OBJECT プロシージャのパラメータ

パラメータ	説明
reference	削除するオブジェクトの参照。

戻り値

なし。

プラグマ

なし。

例外

発生する可能性があります。

## 例

この例は、次のシナリオを実行するための UTL\_REF パッケージの使用法を示しています。会社の従業員が自宅住所の変更を上司に連絡するとします。

... Address\_t の宣言など ...

```
CREATE OR REPLACE TYPE Person_t (
    name    VARCHAR2(64),
    gender  CHAR(1),
    address Address_t,
    MEMBER PROCEDURE setAddress(addr IN Address_t)
);

CREATE OR REPLACE TYPE BODY Person_t (
    MEMBER PROCEDURE setAddress(addr IN Address_t) IS
    BEGIN
        address := addr;
    END;
);
```

```
CREATE OR REPLACE TYPE Employee_t (
```

Person\_t で、REF を使用して Person\_t への継承の実現と setAddress の委任をシミュレーションします。

```
    thePerson REF Person_t,
    empno     NUMBER(5),
    deptREF   Department_t,
    mgrREF    Employee_t,
    reminders StringArray_t,
    MEMBER PROCEDURE setAddress(addr IN Address_t),
    MEMBER procedure addReminder(reminder VARCHAR2);
);

CREATE TYPE BODY Employee_t (
    MEMBER PROCEDURE setAddress(addr IN Address_t) IS
        myMgr Employee_t;
        meAsPerson Person_t;
    BEGIN
```

責任を thePerson に委任して、アドレスを更新します。個人オブジェクトを参照からロックして、それをまた選択します。

```
        UTL_REF.LOCK_OBJECT(thePerson, meAsPerson);
        meAsPerson.setAddress(addr);
```

thePerson に委任します。



```
UTL_REF.UPDATE_OBJECT(thePerson, meAsPerson);  
if mgr is NOT NULL THEN
```

マネージャに覚書きを渡します。

```
    UTL_REF.LOCK_OBJECT(mgr);  
    UTL_REF.SELECT_OBJECT(mgr, myMgr);  
    myMgr.addReminder  
    ('Update address in the employee directory for' ||  
     thePerson.name || ', new address: ' || addr.asString);  
    UTL_REF.UPDATE_OBJECT(mgr, myMgr);  
END IF;  
EXCEPTION  
    WHEN OTHERS THEN  
        errnum := SQLCODE;  
        errmsg := SUBSTR(SQLERRM, 1, 200);
```



**C**

catproc.sql スクリプト, 1-2  
CLOB データ型  
    NCLOB, 17-3  
CREATE PACKAGE BODY コマンド, 1-3  
CREATE PACKAGE コマンド, 1-3

**D**

DBMS\_ALERT パッケージ, 2-1  
DBMS\_APPLICATION\_INFO パッケージ, 3-2  
DBMS\_AQADM パッケージ, 5-1  
DBMS\_AQ パッケージ, 4-1  
DBMS\_DDL パッケージ, 6-1  
DBMS\_DEBUG パッケージ, 7-1  
DBMS\_DEFER\_QUERY パッケージ, 9-1  
DBMS\_DEFER\_SYS パッケージ, 10-1  
DBMS\_DEFER パッケージ, 8-1  
DBMS\_DESCRIBE パッケージ, 11-1  
DBMS\_DISTRIBUTED\_TRUST\_ADMIN パッケージ, 12-1  
DBMS\_HS\_PASSTHROUGH パッケージ, 14-1  
DBMS\_HS パッケージ, 13-1  
DBMS\_IOT パッケージ, 15-1  
DBMS\_JOB パッケージ, 16-1  
DBMS\_LOB パッケージ, 17-1  
DBMS\_LOCK パッケージ, 18-1  
DBMS\_LOGMNR\_D パッケージ, 20-1  
DBMS\_LOGMNR パッケージ, 19-1  
DBMS\_MVIEW パッケージ  
    DBMS\_SNAPSHOT パッケージ, 45-1  
DBMS\_OFFLINE\_OG パッケージ, 21-1  
DBMS\_OFFLINE\_SNAPSHOT パッケージ, 22-1  
DBMS\_OLAP パッケージ, 23-1

DBMS\_ORACLE\_TRACE\_AGENT パッケージ, 24-1  
DBMS\_ORACLE\_TRACE\_USER パッケージ, 25-1  
DBMS\_OUTPUT パッケージ, 26-1  
DBMS\_PCLXUTIL パッケージ, 27-1  
DBMS\_PIPE パッケージ, 28-1  
DBMS\_PROFILER パッケージ, 29-1  
DBMS\_RANDOM パッケージ, 30-1  
DBMS\_RECTIFIER\_DIFF パッケージ, 31-1  
DBMS\_REFRESH パッケージ, 32-1  
DBMS\_REPAIR パッケージ, 33-1  
DBMS\_REPCAT\_ADMIN パッケージ, 35-1  
DBMS\_REPCAT\_INSTANTIATE パッケージ, 36-1  
DBMS\_REPCAT\_RGT パッケージ, 37-1  
DBMS\_REPCAT パッケージ, 34-1  
DBMS\_REPUTIL パッケージ, 38-1  
DBMS\_RESOURCE\_MANAGER\_PRIVS パッケージ, 40-1  
DBMS\_RESOURCE\_MANAGER パッケージ, 39-1  
DBMS\_RLS パッケージ, 41-1  
DBMS\_ROWID パッケージ, 42-1  
DBMS\_SESSION パッケージ, 43-1  
DBMS\_SHARED\_POOL パッケージ, 44-1  
DBMS\_SNAPSHOT パッケージ  
    DBMS\_MVIEW パッケージ, 45-1  
DBMS\_SPACE\_ADMIN パッケージ, 47-1  
DBMS\_SPACE パッケージ, 46-1  
DBMS\_SQL パッケージ  
    エラーの位置, 48-31  
DBMS\_STATS パッケージ, 49-1  
DBMS\_TRACE パッケージ, 50-1  
DBMS\_TRANSACTION パッケージ, 51-1  
DBMS\_TTS パッケージ, 52-1  
DBMS\_UTILITY パッケージ, 53-1  
DEBUG\_EXPTOC パッケージ, 54-1  
DefDefaultDest 表

宛先の削除, 10-4  
宛先の追加, 10-3  
DefError 表  
トランザクションの削除, 10-5  
DESC\_TAB データ型, 48-29

## L

---

LOB  
DBMS\_LOB パッケージ, 17-1

## O

---

OR REPLACE 句  
パッケージの作成, 1-3  
Oracle アドバンスト・キューイング (Oracle AQ)  
DBMS\_AQADM パッケージ, 5-1  
OUTLN\_PKG パッケージ, 55-1

## P

---

PL/SQL  
データ型, 11-6  
数値コード, 11-8

## R

---

RepCatLog ビュー  
削除, 34-60  
RepColumn\_Group 表  
更新, 34-23  
RepGroup ビュー  
更新, 34-25  
RepObject 表  
更新, 34-27  
RepPriority\_Group 表  
更新, 34-24  
RepResolution\_Statistics 表  
削除, 34-61  
RepResolution 表  
更新, 34-28  
RepSite ビュー  
更新, 34-26  
ROWID のデータ型  
拡張形式, 42-12  
DBMS\_ROWID パッケージ, 42-1

## S

---

SDO\_ADMIN パッケージ, 1-15  
SDO\_GEOM パッケージ, 1-15  
SDO\_MIGRATE パッケージ, 1-15  
SDO\_TUNE パッケージ, 1-15  
SQL\*Plus  
順序番号の作成, 1-5  
SQL 文  
32KB を超える, 48-12

## T

---

TimeScale パッケージ, 1-16  
TimeSeries パッケージ, 1-17  
TSTools パッケージ, 1-20

## U

---

UTL\_COLL パッケージ, 56-1  
UTL\_FILE パッケージ, 57-1  
UTL\_HTTP パッケージ, 58-1  
UTL\_PG パッケージ, 1-21  
UTL\_RAW パッケージ, 59-1  
UTL\_REF パッケージ, 60-1

## V

---

Vir\_Pkg パッケージ, 1-22

## あ

---

アドバンスト・キューイング  
DBMS\_AQADM パッケージ, 5-1  
管理インタフェース, 4-11

## い

---

インポート  
オブジェクト・グループ  
オフライン・インスタンスエーション, 21-3,  
21-5  
状態チェック, 34-67  
スナップショット  
オフライン・インスタンスエーション, 22-2,  
22-3

## え

---

### エラー

- DBMS\_ALERT パッケージが戻すエラー , 15-3
- 動的 SQL 内での位置 , 48-31

## お

---

### オフライン・インスタンスーション

- スナップショット , 22-2 , 22-3
- レプリケート・オブジェクト・グループ , 21-2 , 21-3 , 21-4 , 21-5 , 21-6

## か

---

### カーソル

- DBMS\_SQL パッケージ , 48-4

### 各国語サポート

- NCLOB , 17-3

### カレンダー・パッケージ , 1-13

## き

---

### キャラクタ・セット

- ANY\_CS , 17-3

### キューイング

- DBMS\_AQADM パッケージ , 5-1

### 休止中

- レプリケート・スキーマ , 34-72

### 競合

#### 解消

- 追加方法 , 34-10
- 統計 , 34-22 , 34-64

## こ

---

### コレクション

- 表項目 , 48-14

## さ

---

### サイト優先順位

- 変更 , 34-19

### サイト優先順位グループ

- 削除 , 34-50
- 作成
  - 構文 , 34-41

- メンバーの削除 , 34-50

- メンバーの追加 , 34-9

### 削除

- RepCatLog 表 , 34-60
- サイト優先順位グループ , 34-50
- スナップショット・サイト , 34-51
- 統計 , 34-61
- マスター・サイト , 34-66
- 優先順位グループ , 34-47
- 列グループ
  - 構文 , 34-42
- レプリケート・オブジェクト
  - グループ , 34-44
  - スナップショット・サイト , 34-52
  - マスター・サイト , 34-46

### 作成

- サイト優先順位グループ
  - 構文 , 34-41
- スナップショット・サイト
  - 構文 , 34-35
- パッケージ , 1-3
- 優先順位グループ , 34-39
- リフレッシュ・グループ , 32-6
- 列グループ
  - 構文 , 34-38 , 34-59
- レプリケート・オブジェクト
  - 構文 , 34-32
  - サポートの生成 , 34-56
  - スナップショット・サイト , 34-36
- レプリケート・オブジェクト・グループ
  - 構文 , 34-31

## し

---

### 使用禁止

- 伝播 , 10-17

### 実行フロー

- 動的 SQL , 48-4

### 状態

- 伝播 , 10-6

### ジョブ

- キュー
  - ジョブの削除 , 10-19

## す

---

### ストアド・アウトライン

- OUTLN\_PKG パッケージ, 55-1
- スナップショット
  - オフライン・インスタンスエーション, 22-2, 22-3
  - リフレッシュ, 45-7, 45-9, 45-10
- スナップショット・サイト
  - 削除, 34-51
  - 作成
    - 構文, 34-35
  - マスターの変更, 34-73
  - マスターへの変更内容の伝播, 10-15
  - リフレッシュ
    - 構文, 34-62
- スナップショット・ログ
  - マスター表
    - 削除, 45-4, 45-5

## せ

---

- 生成
  - スナップショット・サポート, 34-57

## ち

---

- 遅延トランザクション
  - DefDefaultDest 表
    - 宛先の削除, 10-4
    - 宛先の追加, 10-3
  - 開始, 8-5
  - キューからの削除, 10-5
  - 再実行, 10-8
  - スケジュールの実行, 10-15
  - 遅延リモート・プロシージャ・コール (RPC)
    - 作成, 8-2
    - 即時実行, 10-11
    - 引数, 8-4
    - 引数型, 9-3
    - 引数値, 9-6

- 違い
  - 複数の表, 31-2
  - 調整, 31-5

- 調整
  - 表, 31-5

## て

---

- データ型
  - DBMS\_DESCRIBE, 11-4

- DESC\_TAB, 48-29
- NCLOB, 17-3
- PL/SQL
  - 数値コード, 11-8
- ROWID, 42-1
- データ定義言語 (DDL)
  - 非同期の提供, 34-55
- 伝播
  - 使用禁止, 10-17
  - 状態, 10-6
  - 変更内容
    - 伝播方法の変更, 34-14

## と

---

- 統計
  - 収集, 34-64
- 動的 SQL
  - DBMS\_SQL ファンクション、使用方法, 48-2
  - エラー、位置, 48-31
  - 実行フロー, 48-4
  - 無名ブロック, 48-2

## は

---

- 配列
  - BIND\_ARRAY プロシージャ, 48-6
  - DBMS\_SQL を使用した一括 DML, 48-14
- はじめに
  - 表記規則の例の一覧, xxxiv
- パッケージ
  - 作成, 1-3
  - 参照, 1-5
  - 参照先, 1-6
- パッケージの概要, 1-2

## ひ

---

- 比較
  - 表, 31-2
- 非同期
  - DDL, 34-55
- 表
  - 調整, 31-5
  - 配列としての表項目, 48-14
  - 比較, 31-2

## ふ

---

ファイン・グレイン・アクセス・コントロール  
DBMS\_RLS パッケージ, 41-1  
プランの安定性, 55-1

## へ

---

### 変更

伝播方法, 34-14, 34-21  
優先順位レベル, 34-17  
レプリケート・オブジェクト, 34-15  
変更内容の伝播  
伝播方法の変更, 34-21

## ま

---

マスター・サイト  
削除, 34-66  
作成, 34-7  
変更内容の伝播, 10-15  
マスター定義サイト  
再配置, 34-65

## む

---

無名 PL/SQL ブロック  
動的 SQL, 48-2

## ゆ

---

優先順位グループ  
サイト優先順位グループ  
メンバーの追加, 34-9  
削除, 34-47  
作成, 34-39  
メンバーの削除, 34-47, 34-48  
メンバーの追加, 34-8  
メンバーの変更  
値, 34-18  
優先順位, 34-17

## り

---

リフレッシュ  
スナップショット, 45-7, 45-9, 45-10  
スナップショット・サイト

構文, 34-62  
リフレッシュ・グループ  
削除, 32-5  
新規作成, 32-6  
メンバーを削除, 32-9  
メンバーを追加, 32-2  
リフレッシュ  
手動, 32-8  
リフレッシュ間隔  
変更, 32-3

## れ

---

列グループ  
削除  
構文, 34-42  
作成  
構文, 34-38, 34-59  
メンバーの削除  
構文, 34-43  
メンバーの追加  
構文, 34-5  
レプリケーション・アクティビティの再開, 34-68  
レプリケート・オブジェクト  
DROP\_MASTER\_REPOBJECT, 34-46  
削除  
スナップショット・サイト, 34-52  
作成  
スナップショット・サイト, 34-36  
マスター・サイト, 34-32  
サポートの生成, 34-56  
変更, 34-15  
マスター・グループの作成, 34-31  
レプリケート・オブジェクト・グループ  
オフライン・インスタンスエーション, 21-2, 21-3, 21-4, 21-5, 21-6  
削除, 34-44

