

# Oracle Internet Directory

アプリケーション開発者ガイド

リリース 2.1.1

2000 年 11 月

部品番号 : J02324-01

ORACLE®

---

Oracle Internet Directory アプリケーション開発者ガイド, リリース 2.1.1

部品番号 : J02324-01

原本名 : Oracle Internet Directory Application Developer's Guide, Release 2.1.1

原本部品番号 : A86082-01

原本著者 : Richard Smith

原本協力者 : Saheli Dey, Rajinder Gupta, Ashish Kolli, Stephen Lee, David Lin, Radikha Moolky, David Saslav

Copyright © 1996, 2000, Oracle Corporation. All rights reserved.

Printed in Japan.

制限付権利の説明

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

当プログラムのリバース・エンジニアリング等は禁止されています。

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

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

危険な用途への使用について

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

Restricted Rights Notice

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

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

---

# 目次

はじめに .....	iii
<b>1 概要</b>	
Oracle Internet Directory Software Developer's Kit リリース 2.1.1 の概要 .....	1-2
Oracle Internet Directory Software Developer's Kit のコンポーネント .....	1-2
Oracle Internet Directory のその他のコンポーネント .....	1-2
サポートされるオペレーティング・システム .....	1-3
<b>2 C API</b>	
Oracle Internet Directory C API の概要 .....	2-2
Oracle Internet Directory SDK C API SSL 機能拡張 .....	2-2
LDAP C API の概要 .....	2-3
C API の使用例 .....	2-6
SSL モードでの C API の使用方法 .....	2-6
非 SSL モードでの C API の使用方法 .....	2-7
C API を使用したアプリケーションの作成 .....	2-8
必要なヘッダー・ファイルとライブラリ .....	2-8
サンプル検索ツールの作成 .....	2-8
依存性と制限事項 .....	2-20
<b>3 PL/SQL API</b>	
Oracle Internet Directory PL/SQL API の概要 .....	3-2
PL/SQL の使用例 .....	3-2
データベース・トリガーから PL/SQL API を使用する方法 .....	3-2

PL/SQL API での検索方法 .....	3-9
PL/SQL LDAP API を使用したアプリケーションの作成 .....	3-13
依存性と制限事項 .....	3-13
PL/SQL のリファレンス .....	3-13
サブプログラムの概要 .....	3-14
例外の概要 .....	3-16
データ型の概要 .....	3-18
サブプログラム .....	3-19

## 4 コマンドライン・ツールの構文

LDAP データ交換フォーマット (LDIF) 構文 .....	4-2
コマンドライン・ツールの構文 .....	4-4
ldapadd 構文 .....	4-4
ldapaddmt 構文 .....	4-6
ldapbind 構文 .....	4-7
ldapcompare 構文 .....	4-8
ldapdelete 構文 .....	4-10
ldapmoddn 構文 .....	4-11
ldapmodify 構文 .....	4-12
ldapmodifymt 構文 .....	4-16
ldapsearch 構文 .....	4-17
カタログ管理ツールの構文 .....	4-21

## 索引

---

# はじめに

『Oracle Internet Directory アプリケーション開発者ガイド』では、C アプリケーション・プログラミング・インタフェース (C API) および PL/SQL API を利用して、アプリケーションで Oracle Internet Directory にアクセスする方法について説明します。

## 対象読者

『Oracle Internet Directory アプリケーション開発者ガイド』は、Oracle Internet Directory サーバーに対してディレクトリ情報の格納および更新を行うアプリケーションを作成する開発者を対象としています。また、このマニュアルは、Oracle Internet Directory C API および PL/SQL API の動作の理解が必要な人も対象としています。

## このマニュアルの構成

### 第 1 章「概要」

Oracle Internet Directory Software Developer's Kit リリース 2.1.1 の対象読者とコンポーネントについての概要を説明します。また、Oracle Internet Directory のその他のコンポーネントと、サポートしているプラットフォームのリストを示します。

### 第 2 章「C API」

Oracle Internet Directory C API について説明し、その使用例を紹介します。

### 第 3 章「PL/SQL API」

DBMS\_LDAP という PL/SQL パッケージに同梱されている PL/SQL API について説明します。また、その使用例も紹介します。

### 第 4 章「コマンドライン・ツールの構文」

LDAP データ交換フォーマット (LDIF) と LDAP コマンドライン・ツールを使用するための構文、使用方法および使用例を紹介します。

## 関連ドキュメント

『Oracle Internet Directory 管理者ガイド』

Oracle8i ドキュメント・セット

『Chadwick, David, Understanding X.500 The Directory. Thomson Computer Press, 1996』  
この書籍は現在絶版になっていますが、次の Web サイトからオンライン版を入手できます。  
<http://www.salford.ac.uk/its024/Version.Web/Contents.htm>

Hodges, Jeff, Staff Scientist, Oblix, Inc.,  
<http://www.kingsmountain.com/ldapRoadmap.shtml>

『Howes, Tim and Mark Smith, LDAP: Programming Directory-enabled Applications with Lightweight Directory Access Protocol. Macmillan Technical Publishing, 1997』

『Howes, Tim, Mark Smith and Gordon Good, Understanding and Deploying LDAP Directory Services. Macmillan Technical Publishing, 1999』

『Kosiur, Dave, LDAP: "The next-generation directory?," SunWorld Online, October 1997』

『Radicati, Sara, X.500 Directory Services, Technology and Deployment, International Thomson Computer Press, 1994』

University of Michigan LDAP Repository,  
<http://www.umich.edu/~dirsvcs/ldap/index.html>

# 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
. . .	縦の省略記号が例の中で使用されている場合は、例に直接関係のない情報が省略されていることを示します。
...	横の省略記号が文またはコマンド内で使用されている場合は、その文またはコマンドの一部が省略されていることを示します。
太字	太字のテキストは、コマンドでユーザーが入力する必要があるテキスト、または小見出しを示します。
イタリック	イタリック体は、コード例の中でユーザーが値を指定する必要がある変数を示します。
固定幅フォント	固定幅フォントは、ユーザーの入力例とコード例に使用されます。
構文	この字体は、コード例の中の構文説明に使用されます。
<>	山カッコは、コード例の中で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構いません。
{ }	中カッコは選択が必須の項目を表します。選択肢の中から1つ選択します。



この章では、Oracle Internet Directory Software Developer's Kit リリース 2.1.1 の対象読者とコンポーネントについての概要を説明します。また、Oracle Internet Directory のその他のコンポーネントと、サポートしているプラットフォームのリストを示します。

この章では、次の項目について説明します。

- [Oracle Internet Directory Software Developer's Kit リリース 2.1.1 の概要](#)
- [Oracle Internet Directory Software Developer's Kit のコンポーネント](#)
- [Oracle Internet Directory のその他のコンポーネント](#)
- [サポートされるオペレーティング・システム](#)

## Oracle Internet Directory Software Developer's Kit リリース 2.1.1 の概要

Oracle Internet Directory SDK リリース 2.1.1 は、C、C++ および PL/SQL を使用するアプリケーション開発者を対象とした製品です。Java 開発者は、Sun の JNDI プロバイダを使用して、Oracle Internet Directory サーバー内のディレクトリ情報にアクセスしてください。

## Oracle Internet Directory Software Developer's Kit のコンポーネント

Oracle Internet Directory Software Developer's Kit リリース 2.1.1 の構成内容は次のとおりです。

- LDAP バージョン 2 に準拠した C アプリケーション・プログラミング・インタフェース (C API)
- PL/SQL API (DBMS\_LDAP という PL/SQL パッケージに同梱)
- サンプル・プログラム
- 『Oracle Internet Directory アプリケーション開発者ガイド』 (このマニュアル)
- コマンドライン・ツール

## Oracle Internet Directory のその他のコンポーネント

次に示す Oracle Internet Directory リリース 2.1.1 のコンポーネントは、Oracle Internet Directory Software Developer's Kit の一部ではありませんが、個別に取得できます。

- Oracle Internet Directory サーバー (LDAP バージョン 3 に準拠したディレクトリ・サーバー)
- Oracle Directory Replication Server
- Oracle Directory Manager (Java ベースのグラフィカル・ユーザー・インタフェース)
- Oracle Internet Directory バルク・ツール
- 『Oracle Internet Directory 管理者ガイド』
- 『Oracle Internet Directory インストレーション・ガイド』

## サポートされるオペレーティング・システム

Oracle Internet Directory は、サーバー、クライアントともに、次のオペレーティング・システムをサポートしています。

- Sun Solaris
- Microsoft Windows
  - Windows NT 4.0
  - Windows 95
  - Windows 98
  - Windows 2000
- HP-UX
- AIX
- Compaq TRU64
- Intel Solaris
- SGI
- DGUX
- UNIXWARE



この章では、Oracle Internet Directory C アプリケーション・プログラミング・インタフェース（C API）について説明し、その使用例を紹介します。この章では、次の項目について説明します。

- [Oracle Internet Directory C API の概要](#)
- [C API の使用例](#)
- [C API を使用したアプリケーションの作成](#)
- [依存性と制限事項](#)

## Oracle Internet Directory C API の概要

Oracle Internet Directory SDK C API リリース 2.1.1 は、次のものをベースにしています。

- LDAP バージョン 2 C API
- SSL をサポートするための Oracle の機能拡張

Oracle Internet Directory SDK C API リリース 2.1.1 は、次のモードをサポートしています。

- SSL モード SSL を使用してすべての通信を保護する
- 非 SSL モードクライアントからサーバーへの通信を保護しない

SSL モードを使用するには、Oracle SSL コール・インタフェースを使用する必要があります。API を使用する際に、SSL コールの有無によってどちらのモードを使用するかを決定します。SSL モードと非 SSL モードは簡単に切り替えることができます。

**関連項目：** この 2 つのモードの使用方法についての詳細は、2-6 ページの「[C API の使用例](#)」を参照してください。

この項では、次の項目について説明します。

- [Oracle Internet Directory SDK C API SSL 機能拡張](#)
- [LDAP C API の概要](#)

## Oracle Internet Directory SDK C API SSL 機能拡張

LDAP API への Oracle SSL 機能拡張は、標準的な SSL プロトコルに基づいています。SSL 機能拡張は回線を通るデータの暗号化と復号化および認証を行います。

認証には次の 3 つのモードがあります。

- サーバー認証—クライアントによるサーバーの認証のみ
- クライアントとサーバーの認証—クライアントとサーバーが相互に認証
- 認証なし—クライアントとサーバーのどちらも認証せず、SSL による暗号化のみ使用

認証のタイプは、SSL インタフェース・コールのパラメータで指定します。

### SSL インタフェース・コール

次のコールを行うだけで、SSL は使用可能になります。

```
int ldap_init_SSL(Sockbuf *sb, text *sslwallet, text *sslwalletpasswd, int
sslauthmode)
```

ldap\_init\_SSL コールは、標準的な SSL プロトコルを使用して、クライアントとサーバーの間で必要なハンドシェイクを実行します。コールが成功すると、これ以降のすべての通信は保護された接続で実行されます。

引数	説明
sb	LDAP ハンドルの一部として、 <i>ldap_open</i> コールによって戻されたソケット・バッファ・ハンドル。
sslwallet	ユーザー Wallet の位置。
sslwalletpasswd	Wallet を使用するために必要なパスワード。
sslauthmode	<p>ユーザーが使用できる SSL 認証モード。使用できる値は次のとおりです。</p> <ul style="list-style-type: none"> <li>■ GSLC_SSL_NO_AUTH – 認証の必要なし</li> <li>■ GSLC_SSL_ONEWAY_AUTH – サーバー認証のみ必要</li> <li>■ GSLC_SSL_TWOWAY_AUTH – クライアントとサーバーの両方の認証が必要</li> </ul> <p>戻り値が 0（ゼロ）のときは、処理は成功です。戻り値が 0（ゼロ）以外のときは、エラーです。エラー・コードは、<i>ldap_err2string</i> ファクションを使用してエラー・メッセージに変換できます。</p>

**関連項目：** 2-6 ページの「[C API の使用例](#)」を参照してください。

## Wallet サポート

SSL の機能を使用するには、使用している認証モードに応じて、サーバーとクライアントそれぞれに Wallet が必要になります。この C API リリース 2.1.1 では Oracle Wallet のみをサポートしています。Oracle Wallet Manager を使用して Wallet を作成できます。

## LDAP C API の概要

この項では、RFC 1823 に規定されている LDAP C API で使用可能なすべてのコールを示します。

**関連項目：** 次の URL を参照してください。  
<http://www.ietf.org/rfc/rfc1823.txt>（各コールの詳細な説明）

この項では、次の項目について説明します。

- [LDAP セッションの開始と終了](#)
- [LDAP サーバーへの認証](#)
- [LDAP 操作の実行](#)

- [検索結果の取得](#)
- [識別名の操作](#)
- [エラー操作](#)
- [メモリーの解放](#)

## LDAP セッションの開始と終了

<code>ldap_open()</code>	LDAP サーバーへの接続のオープン
<code>ldap_unbind()</code>	LDAP セッションの終了

## LDAP サーバーへの認証

<code>ldap_bind()</code>	LDAP サーバーへの一般認証
<code>ldap_bind_s()</code>	
<code>ldap_simple_bind()</code>	LDAP サーバーへの簡易認証
<code>ldap_simple_bind_s()</code>	

## LDAP 操作の実行

<code>ldap_add()</code> / <code>ldap_add_s()</code>	ディレクトリに新規エントリを追加
<code>ldap_modify()</code> / <code>ldap_modify_s()</code>	ディレクトリ内のエントリの変更
<code>ldap_delete()</code> / <code>ldap_delete_s()</code>	ディレクトリからのエントリの削除
<code>ldap_modrdn()</code> / <code>ldap_modrdn_s()</code>	ディレクトリ内のエントリの相対識別名 (RDN) を変更
<code>ldap_search()</code> / <code>ldap_search_s()</code>	ディレクトリの検索
<code>ldap_search_st()</code>	タイムアウト値でのディレクトリの検索
<code>ldap_compare()</code> / <code>ldap_compare_s()</code>	ディレクトリ内のエントリの比較
<code>ldap_result()</code>	非同期操作の結果のチェック
<code>ldap_abandon()</code>	非同期操作の取消し

## 検索結果の取得

<code>ldap_get_dn()</code>	エントリの識別名の取得
<code>ldap_first_entry()</code>	一連の検索結果から最初のエントリを取得
<code>ldap_next_entry()</code>	一連の検索結果から次のエントリを取得
<code>ldap_count_entries()</code>	一連の検索結果のエントリ件数をカウント
<code>ldap_first_attribute()</code>	エントリ内の最初の属性の名前を取得
<code>ldap_next_attribute()</code>	エントリ内の次の属性の名前を取得
<code>ldap_get_values()</code>	属性の文字列値を取得
<code>ldap_get_values_len()</code>	属性のバイナリ値を取得
<code>ldap_count_values()</code>	属性の文字列値をカウント
<code>ldap_count_values_len()</code>	属性のバイナリ値をカウント

## 識別名の操作

<code>ldap_get_dn()</code>	エントリの識別名の取得
<code>ldap_explode_dn()</code>	識別名を構成要素に分割
<code>ldap_dn2ufn()</code>	ユーザーにわかりやすい名前に変換

## エラー操作

<code>ldap_result2error()</code>	結果メッセージからエラー・コードを取得
<code>ldap_err2string()</code>	特定のエラー・コードのエラー・メッセージを取得
<code>ldap_perror</code>	エラー・メッセージの中から指定したメッセージを出力

## メモリーの解放

<code>ldap_memfree()</code>	LDAP API ファンクション・コールによって割り当てられたメモリーの解放
<code>ldap_msgfree()</code>	検索結果あるいは他の LDAP 操作結果のために割り当てられたメモリーの解放
<code>ldap_value_free()</code>	属性の文字列値のために割り当てられたメモリーの解放

<code>ldap_value_free_len()</code>	属性のバイナリ値のために割り当てられたメモリの解放
<code>ber_free()</code>	BerElement 構造体のために割り当てられたメモリの解放

## C API の使用例

SSL モードおよび非 SSL モードでの API の使用例を次に示します。詳細な例は、RFC 1823 に記載されています。また、LDAP 検索を実行するコマンドライン・ツールのためのサンプル・コードも、それぞれのモードでの API の使用方法を示しています。

この項では、次の項目について説明します。

- [SSL モードでの C API の使用方法](#)
- [非 SSL モードでの C API の使用方法](#)

## SSL モードでの C API の使用方法

```
#include <stdio.h>
#include <string.h>
#include <ctype.h>
#include <netdb.h>
#include <gsle.h>
#include <gslc.h>
#include <gsld.h>
#include "gsldcc.h"

main()
{
    LDAP          *ld;
    int           ret = 0;
    ....
    /* open a connection */
    if ( (ld = ldap_open( "MyHost", 636 )) == NULL )
        exit( 1 );

    /* SSL initialization */
    ret = ldap_init_SSL(&ld->ld_sb, "file:/sslwallet", "welcome",
                       GSLC_SSL_ONETIME_AUTH );

    if(ret != 0)
    {
        printf(" %s \n", ldap_err2string(ret));
        exit(1);
    }
}
```

```

/* authenticate as nobody */
if ( ldap_bind_s( ld, NULL, NULL ) != LDAP_SUCCESS ) {
    ldap_perror( ld, "ldap_bind_s" );
    exit( 1 );
}

....
....
}

```

ldap\_init\_SSL をコールしているので、この例でのクライアントからサーバーへの通信は、SSL を使用することによって保護されています。

## 非 SSL モードでの C API の使用方法

```

#include <stdio.h>
#include <string.h>
#include <ctype.h>
#include <netdb.h>
#include <gsle.h>
#include <gslc.h>
#include <gsld.h>
#include "gslcc.h"

main()
{
    LDAP          *ld;
    int            ret = 0;
    ....

    /* open a connection */
    if ( (ld = ldap_open( "MyHost", LDAP_PORT )) == NULL )
        exit( 1 );

    /* authenticate as nobody */
    if ( ldap_bind_s( ld, NULL, NULL ) != LDAP_SUCCESS ) {
        ldap_perror( ld, "ldap_bind_s" );
        exit( 1 );
    }
    ....
    ....
}

```

この例では、ldap\_init\_SSL をコールしていないので、クライアントからサーバーへの通信は保護されていません。

## C API を使用したアプリケーションの作成

この項では、次の項目について説明します。

- 必要なヘッダー・ファイルとライブラリ
- サンプル検索ツールの作成

### 必要なヘッダー・ファイルとライブラリ

C API を使用してアプリケーションを作成するには、次のものが必要となります。

- \$ORACLE\_HOME/ldap/public/ldap.h ヘッダー・ファイル
- \$ORACLE\_HOME/lib/libldapclnt8.a ライブラリ

### サンプル検索ツールの作成

Oracle Internet Directory SDK リリース 2.1.1 は、C API を使用したアプリケーションの作成方法を説明するために、samplesearch というサンプル・コマンドライン・ツールを提供しています。samplesearch を使用すると、SSL モードおよび非 SSL モードで LDAP 検索を実行できます。

Source ファイル (samplesearch.c) と Make ファイル (demo\_ldap.mk) はディレクトリ ORACLE\_HOME/ldap/demo にあります。

サンプル検索ツールを作成するには、次のコマンドを入力します。

```
make -f demo_ldap.mk build EXE=samplesearch OBJS=samplesearch.o
```

---

---

**注意：** この Make ファイルは、C API を使用して他のクライアント・アプリケーションを作成するときにも使用できます。samplesearch を作成するバイナリ・ファイルの名前に、samplesearch.o をオブジェクト・ファイルに置き換えてください。

---

---

ldapsearch のサンプル・コードは次のとおりです。

```
/*
NAME
    s0gsldsearch.c - <one-line expansion of the name>
DESCRIPTION
    <short description of component this file declares/defines>
PUBLIC FUNCTION(S)
    <list of external functions declared/defined - with one-line descriptions>
PRIVATE FUNCTION(S)
    <list of static functions defined in .c file - with one-line descriptions>
```

```

RETURNS
    <function return values, for .c file with single function>
NOTES
    <other useful comments, qualifications, etc.>
*/
#include <stdio.h>
#include <string.h>
#include <ctype.h>
#include <netdb.h>
#include "ldap.h"

#define DEFSEP="
#define LDAPSEARCH_BINDDN      NULL
#define LDAPSEARCH_BASE        DEFAULT_BASE
#define DEFAULT_BASE           "o=oracle, c=US"

#ifdef LDAP_DEBUG
extern int ldap_debug, lber_debug;
#endif /* LDAP_DEBUG */

usage( s )
char*s;
{
    fprintf( stderr, "usage: %s [options] filter [attributes...]\nwhere:\n", s );
    fprintf( stderr, "    filter\tRFC-1558 compliant LDAP search filter\n" );
    fprintf( stderr, "    attributes\twhitespace-separated list of attributes to
retrieve\n" );
    fprintf( stderr, "\t\t(if no attribute list is given, all are retrieved)\n" );
    fprintf( stderr, "options:\n" );
    fprintf( stderr, "    -n\t\tshow what would be done but don't actually search\n"
);
    fprintf( stderr, "    -v\t\ttrun in verbose mode (diagnostics to standard
output)\n" );
    fprintf( stderr, "    -t\t\twrite values to files in /tmp\n" );
    fprintf( stderr, "    -u\t\tinclude User Friendly entry names in the output\n"
);
    fprintf( stderr, "    -A\t\tretrieve attribute names only (no values)\n" );
    fprintf( stderr, "    -B\t\tdo not suppress printing of non-ASCII values\n" );
    fprintf( stderr, "    -I\t\tprint entries in LDIF format (-B is implied)\n" );
#ifdef LDAP_REFERRALS
    fprintf( stderr, "    -R\t\tdo not automatically follow referrals\n" );
#endif /* LDAP_REFERRALS */
    fprintf( stderr, "    -d level\tset LDAP debugging level to `level'\n" );
    fprintf( stderr, "    -F sep\tprint `sep' instead of `=' between attribute names
and values\n" );
    fprintf( stderr, "    -S attr\tsort the results by attribute `attr'\n" );
    fprintf( stderr, "    -f file\tperform sequence of searches listed in `file'\n"

```

```
);
    fprintf( stderr, "    -b basedn\tbase dn for search\n" );
    fprintf( stderr, "    -s scope\tone of base, one, or sub (search scope)\n" );
    fprintf( stderr, "    -a deref\tone of never, always, search, or find (alias
dereferencing)\n" );
    fprintf( stderr, "    -l time lim\ttime limit (in seconds) for search\n" );
    fprintf( stderr, "    -z size lim\tsize limit (in entries) for search\n" );
    fprintf( stderr, "    -D binddn\tbind dn\n" );
    fprintf( stderr, "    -w passwd\tbind passwd (for simple authentication)\n" );
#ifdef KERBEROS
    fprintf( stderr, "    -k\t\tuse Kerberos instead of Simple Password
authentication\n" );
#endif
    fprintf( stderr, "    -h host\tldap server\n" );
    fprintf( stderr, "    -p port\tport on ldap server\n" );
    fprintf( stderr, "    -W Wallet\tWallet location\n" );
    fprintf( stderr, "    -P Wpasswd\tWallet Password\n" );
    fprintf( stderr, "    -U SSLAuth\tSSL Authentication Mode\n" );
    return;
}

static char*binddn = LDAPSEARCH_BINDDN;
static char*passwd = NULL;
static char*base = LDAPSEARCH_BASE;
static char*ldaphost = NULL;
static intldapport = LDAP_PORT;
static char*sep = DEFSEP;
static char*sortattr = NULL;
static intskipsortattr = 0;
static intverbose, not, includeufn, allow_binary, vals2tmp, ldif;
/* TEMP */

main( argc, argv )
intargc;
char**argv;
{
    char*infile, *filtpattern, **attrs, line[ BUFSIZ ];
    FILE*fp;
    intrc, i, first, scope, kerberos, deref, attrsonly;
    intldap_options, timelimit, sizelimit, authmethod;
    LDAP*ld;
    extern char*optarg;
    extern intoptind;
    charlocalHostName[MAXHOSTNAMELEN + 1];
    char *sslwrl = NULL;
    char*sslpaswd = NULL;
    int sslauth=0,err=0;
```

```

        infile = NULL;
        deref = verbose = allow_binary = not = kerberos = vals2tmp =
        attrsonly = ldif = 0;
#ifdef LDAP_REFERRALS
        ldap_options = LDAP_OPT_REFERRALS;
#else /* LDAP_REFERRALS */
        ldap_options = 0;
#endif /* LDAP_REFERRALS */
        sizelimit = timelimit = 0;
        scope = LDAP_SCOPE_SUBTREE;

        while (( i = getopt( argc, argv,
#ifdef KERBEROS
            "KkmuvtrABLD:s:f:h:b:d:p:F:a:w:l:z:S:"
#else
            "muvtrABLD:s:f:h:b:d:p:F:a:w:l:z:S:W:P:U:"
#endif
            )) != EOF ) {
        switch( i ) {
        case 'n':/* do Not do any searches */
            ++not;
            break;
        case 'v':/* verbose mode */
            ++verbose;
            break;
        case 'd':
#ifdef LDAP_DEBUG
            ldap_debug = lber_debug = atoi( optarg );/* */
#else /* LDAP_DEBUG */
            fprintf( stderr, "compile with -DLAP_DEBUG for debugging\n" );
#endif /* LDAP_DEBUG */
            break;
#ifdef KERBEROS
        case 'k':/* use kerberos bind */
            kerberos = 2;
            break;
        case 'K':/* use kerberos bind, 1st part only */
            kerberos = 1;
            break;
#endif
        case 'u':/* include UFN */
            ++includeufn;
            break;
        case 't':/* write attribute values to /tmp files */
            ++vals2tmp;
            break;

```

```
case 'R':/* don't automatically chase referrals */
#ifdef LDAP_REFERRALS
    ldap_options &= ~LDAP_OPT_REFERRALS;
#else /* LDAP_REFERRALS */
    fprintf( stderr,
        "compile with -DLDAF_REFERRALS for referral support\n" );
#endif /* LDAP_REFERRALS */
    break;
case 'A':/* retrieve attribute names only -- no values */
    ++attrsonly;
    break;
case 'L':/* print entries in LDIF format */
    ++ldif;
    /* fall through -- always allow binary when outputting LDIF */
case 'B':/* allow binary values to be printed */
    ++allow_binary;
    break;
case 's':/* search scope */
    if ( strcmp( optarg, "base", 4 ) == 0 ) {
scope = LDAP_SCOPE_BASE;
    } else if ( strcmp( optarg, "one", 3 ) == 0 ) {
scope = LDAP_SCOPE_ONELEVEL;
    } else if ( strcmp( optarg, "sub", 3 ) == 0 ) {
scope = LDAP_SCOPE_SUBTREE;
    } else {
fprintf( stderr, "scope should be base, one, or sub\n" );
usage( argv[ 0 ] );
        exit(1);
    }
    break;

case 'a':/* set alias deref option */
    if ( strcmp( optarg, "never", 5 ) == 0 ) {
deref = LDAP_DEREF_NEVER;
    } else if ( strcmp( optarg, "search", 5 ) == 0 ) {
deref = LDAP_DEREF_SEARCHING;
    } else if ( strcmp( optarg, "find", 4 ) == 0 ) {
deref = LDAP_DEREF_FINDING;
    } else if ( strcmp( optarg, "always", 6 ) == 0 ) {
deref = LDAP_DEREF_ALWAYS;
    } else {
fprintf( stderr, "alias deref should be never, search, find, or always\n" );
usage( argv[ 0 ] );
        exit(1);
    }
    break;
```

```
case 'F':/* field separator */
    sep = (char *)strdup( optarg );
    break;
case 'f':/* input file */
    infile = (char *)strdup( optarg );
    break;
case 'h':/* ldap host */
    ldaphost = (char *)strdup( optarg );
    break;
case 'b':/* searchbase */
    base = (char *)strdup( optarg );
    break;
case 'D':/* bind DN */
    binddn = (char *)strdup( optarg );
    break;
case 'p':/* ldap port */
    ldapport = atoi( optarg );
    break;
case 'w':/* bind password */
    passwd = (char *)strdup( optarg );
    break;
case 'l':/* time limit */
    timelimit = atoi( optarg );
    break;
case 'z':/* size limit */
    sizelimit = atoi( optarg );
    break;
case 'S':/* sort attribute */
    sortattr = (char *)strdup( optarg );
    break;
case 'W':/* Wallet URL */
    sslwrl = (char *)strdup( optarg );
    break;
case 'P':/* Wallet password */
    sslpasswd = (char *)strdup( optarg );
    break;
case 'U':/* SSL Authentication Mode */
    sslauth = atoi( optarg );
    break;
default:
    usage( argv[0] );
    exit(1);
    break;
}

}

if ( argc - optind < 1 ) {
```

```
usage( argv[ 0 ] );
    exit(1);
}
filtpattern = (char *)strdup( argv[ optind ] );
if ( argv[ optind + 1 ] == NULL ) {
attrs = NULL;
} else if ( sortattr == NULL || *sortattr == '\0' ) {
    attrs = &argv[ optind + 1 ];
} else {
for ( i = optind + 1; i < argc; i++ ) {
    if ( strcasecmp( argv[ i ], sortattr ) == 0 ) {
break;
    }
}
if ( i == argc ) {
skipsortattr = 1;
argv[ optind ] = sortattr;
} else {
optind++;
}
    attrs = &argv[ optind ];
}

if ( infile != NULL ) {
if ( infile[0] == '-' && infile[1] == '\0' ) {
    fp = stdin;
} else if ( ( fp = fopen( infile, "r" ) ) == NULL ) {
    perror( infile );
    exit( 1 );
}
}

if ( ldaphost == NULL ) {
    if ( gethostname( localHostName, MAXHOSTNAMELEN ) != 0 ) {
        perror( "gethostname" );
        exit(1);
    }
    ldaphost = localHostName;
}

if ( verbose ) {
printf( "ldap_open( %s, %d )\n", ldaphost, ldapport );
}

if ( ( ld = ldap_open( ldaphost, ldapport ) ) == NULL ) {
perror( ldaphost );
exit( 1 );
}
```

```

    }

    if (sslauth > 1)
    {
        if (!sslwrl || !sslpasswd)
        {
            printf ("Null Wallet or password given\n");
            exit (0);
        }
    }
    if (sslauth > 0)
    {
        if (sslauth == 1)
            sslauth = GSLC_SSL_NO_AUTH;
        else if (sslauth == 2)
            sslauth = GSLC_SSL_ONEWAY_AUTH;
        else if (sslauth == 3)
            sslauth = GSLC_SSL_TWOWAY_AUTH;
        else
        {
            printf(" Wrong SSL Authentication Mode Value\n");
            exit(0);
        }

        err = ldap_init_SSL(&ld->ld_sb,sslwrl,sslpasswd,sslauth);
        if(err != 0)
        {
            printf(" %s\n", ldap_err2string(err));
            exit(0);
        }
    }

    ld->ld_deref = deref;
    ld->ld_timelimit = timelimit;
    ld->ld_sizelimit = sizelimit;
    ld->ld_options = ldap_options;

    if ( !kerberos ) {
authmethod = LDAP_AUTH_SIMPLE;
    } else if ( kerberos == 1 ) {
authmethod = LDAP_AUTH_KRBV41;
    } else {
authmethod = LDAP_AUTH_KRBV4;
    }
    if ( ldap_bind_s( ld, binddn, passwd, authmethod ) != LDAP_SUCCESS ) {
ldap_perror( ld, "ldap_bind" );
exit( 1 );

```

```
    }

    if ( verbose ) {
printf( "filter pattern: %s\nreturning: ", filtpattern );
if ( attrs == NULL ) {
    printf( "ALL" );
} else {
    for ( i = 0; attrs[ i ] != NULL; ++i ) {
printf( "%s ", attrs[ i ] );
    }
}
putchar( '\n' );
}

    if ( infile == NULL ) {
rc = dosearch( ld, base, scope, attrs, attrsonly, filtpattern, "" );
    } else {
rc = 0;
first = 1;
while ( rc == 0 && fgets( line, sizeof( line ), fp ) != NULL ) {
    line[ strlen( line ) - 1 ] = '\0';
    if ( !first ) {
putchar( '\n' );
    } else {
first = 0;
    }
    rc = dosearch( ld, base, scope, attrs, attrsonly, filtpattern,
line );
}
if ( fp != stdin ) {
    fclose( fp );
}
}

    ldap_unbind( ld );
    exit( rc );
}

dosearch( ld, base, scope, attrs, attrsonly, filt patt, value )
LDAP*ld;
char*base;
intscope;
char**attrs;
intattrsonly;
char*filt patt;
char*value;
{
```

```

    charfilter[ BUFSIZ ], **val;
    intrc, first, matches;
    LDAPMessage*res, *e;

    sprintf( filter, filtpatt, value );

    if ( verbose ) {
printf( "filter is: (%s)\n", filter );
    }

    if ( not ) {
return( LDAP_SUCCESS );
    }

    if ( ldap_search( ld, base, scope, filter, attrs, attrsonly ) == -1 ) {
ldap_perror( ld, "ldap_search" );
return( ld->ld_errno );
    }

    matches = 0;
    first = 1;
    while ( (rc = ldap_result( ld, LDAP_RES_ANY, sortattr ? 1 : 0, NULL, &res ))
== LDAP_RES_SEARCH_ENTRY ) {
matches++;
e = ldap_first_entry( ld, res );
if ( !first ) {
    putchar( '\n' );
} else {
    first = 0;
}
print_entry( ld, e, attrsonly );
ldap_msgfree( res );
    }
    if ( rc == -1 ) {
ldap_perror( ld, "ldap_result" );
return( rc );
    }
    if ( ( rc = ldap_result2error( ld, res, 0 ) ) != LDAP_SUCCESS ) {
        ldap_perror( ld, "ldap_search" );
    }
    if ( sortattr != NULL ) {
extern intstrcasecmp();

(void) ldap_sort_entries( ld, &res,
( *sortattr == '\0' ) ? NULL : sortattr, strcasecmp );
matches = 0;
first = 1;

```

```
        for ( e = ldap_first_entry( ld, res ); e != NULLMSG;
              e = ldap_next_entry( ld, e ) ) {
matches++;
        if ( !first ) {
            putchar( '\n' );
        } else {
            first = 0;
        }
        print_entry( ld, e, attrsonly );
    }

    if ( verbose ) {
        printf( "%d matches\n", matches );
    }

    ldap_msgfree( res );
    return( rc );
}
```

```
print_entry( ld, entry, attrsonly )
    LDAP*ld;
    LDAPMessage*entry;
    intattrsonly;
{
    char*a, *dn, *ufn, tmpfname[ 64 ];
    inti, j, notascii;
    BerElement*ber;
    struct berval**bvals;
    FILE*tmpfp;
    extern char*mktemp();

    dn = ldap_get_dn( ld, entry );
    if ( ldif ) {
        write_ldif_value( "dn", dn, strlen( dn ));
    } else {
        printf( "%s\n", dn );
    }
    if ( includeufn ) {
        ufn = ldap_dn2ufn( dn );
        if ( ldif ) {
            write_ldif_value( "ufn", ufn, strlen( ufn ));
        } else {
            printf( "%s\n", ufn );
        }
    }
    free( ufn );
}
```

```

    }
    free( dn );

    for ( a = ldap_first_attribute( ld, entry, &ber ); a != NULL;
          a = ldap_next_attribute( ld, entry, ber ) ) {
    if ( skipsortattr && strcasecmp( a, sortattr ) == 0 ) {
        continue;
    }
    if ( attrsonly ) {
        if ( ldif ) {
            write_ldif_value( a, "", 0 );
        } else {
            printf( "%s\n", a );
        }
    } else if ( ( bvals = ldap_get_values_len( ld, entry, a ) ) != NULL ) {
        for ( i = 0; bvals[i] != NULL; i++ ) {
            if ( vals2tmp ) {
                sprintf( tmpfname, "/tmp/ldapsearch-%s-XXXXXX", a );
                tmpfp = NULL;

                if ( mktemp( tmpfname ) == NULL ) {
                    perror( tmpfname );
                } else if ( ( tmpfp = fopen( tmpfname, "w" ) ) == NULL ) {
                    perror( tmpfname );
                } else if ( fwrite( bvals[ i ]->bv_val,
                                     bvals[ i ]->bv_len, 1, tmpfp ) == 0 ) {
                    perror( tmpfname );
                } else if ( ldif ) {
                    write_ldif_value( a, tmpfname, strlen( tmpfname ) );
                } else {
                    printf( "%s%s\n", a, sep, tmpfname );
                }
            }

            if ( tmpfp != NULL ) {
                fclose( tmpfp );
            }
        } else {
            notascii = 0;
            if ( !allow_binary ) {
                for ( j = 0; j < bvals[ i ]->bv_len; ++j ) {
                    if ( !isascii( bvals[ i ]->bv_val[ j ] ) ) {
                        notascii = 1;
                        break;
                    }
                }
            }
        }
    }
}

```

```
        if ( ldif ) {
write_ldif_value( a, bvals[ i ]->bv_val,
bvals[ i ]->bv_len );
        } else
        {
printf( "%s%s%s\n", a, sep,
notascii ? "NOT ASCII" : (char *)bvals[ i ]->bv_val );
        }
    }
    }
    gsledePBerBvecfree( bvals );
}
}
}

int
write_ldif_value( char *type, char *value, unsigned long vallen )
{
    char *ldif;

    if (( ldif = gsldlDLdifTypeAndValue( type, value, (int)vallen )) == NULL )
    {
        return( -1 );
    }

    fputs( ldif, stdout );
    free( ldif );

    return( 0 );
}
```

## 依存性と制限事項

この API は、どのリリースの Oracle Internet Directory サーバーに対しても、また、サード・パーティ製の LDAP サーバーに対しても、動作が可能です。

SSL で別の認証モードを使用するには、ディレクトリ・サーバーの構成設定をモードに応じて変更する必要があります。

**関連項目：** それぞれの SSL 認証モードに応じた Oracle Internet Directory サーバーの設定方法の詳細は、『Oracle Internet Directory 管理者ガイド』を参照してください。

SSL モードで C API を使用する場合は、Wallet を作成するために Oracle Wallet Manager が必要となります。

TCP/IP ソケット・ライブラリが必要です。

次の Oracle ライブラリが必要です。

- Oracle SSL 関連ライブラリ
- Oracle システム・ライブラリ

サンプル・コマンドライン・ツールのリリースの中には、サンプル・ライブラリが含まれています。それらのライブラリはユーザー自身のライブラリに置き換える必要があります。

この製品は、LDAP SDK の仕様（RFC 1823）に記述されている認証方法のみをサポートします。



---

## PL/SQL API

この章では、Oracle Internet Directory PL/SQL アプリケーション・プログラミング・インタフェース（PL/SQL API）について説明し、その使用例を紹介します。この章では、次の項目について説明します。

- [Oracle Internet Directory PL/SQL API の概要](#)
- [PL/SQL の使用例](#)
- [PL/SQL LDAP API を使用したアプリケーションの作成](#)
- [依存性と制限事項](#)
- [PL/SQL のリファレンス](#)

## Oracle Internet Directory PL/SQL API の概要

Oracle Internet Directory PL/SQL API は、DBMS\_LDAP という PL/SQL パッケージに含まれています。このパッケージを使用すると、PL/SQL アプリケーションで、エンタープライズ・ワイドの LDAP サーバー内にあるデータへアクセスできます。ファンクション・コールのネーミングと構文は、Oracle Internet Directory C API ファンクションに類似しています。ただし、PL/SQL API に含まれるのは、C API で使用できる関数の一部のみです。特に、PL/SQL API で使用できるのは、LDAP サーバーへの同期コールのみです。

## PL/SQL の使用例

この項では、次の項目について説明します。

- [データベース・トリガーから PL/SQL API を使用する方法](#)
- [PL/SQL API での検索方法](#)

## データベース・トリガーから PL/SQL API を使用する方法

DBMS\_LDAP API をデータベース・トリガーからコールすると、データベースの表に加えられた変更とエンタープライズ・ワイドの LDAP サーバーを同期化することができます。次の例は、挿入、更新および削除のトリガーを使用して、EMP という表に加えられた変更を LDAP サーバーと同期化する方法を示したものです。この例に関係するファイルは、trigger.sql および empdata.sql の 2 つです。

ファイル trigger.sql では、表およびその表に関連付けられたトリガーが作成されます。

ファイル empdata.sql では、表 EMP にサンプル・データが挿入されます。表 EMP は、挿入トリガーにより、LDAP サーバーに対して自動的に更新されます。

これらの 2 つのファイルは、\$ORACLE\_HOME/ldap/demo の下位の plsql ディレクトリにあります。

```
$Header: $
Copyright (c) Oracle Corporation 2000. All Rights Reserved.
FILE
trigger.sql
DESCRIPTION
This SQL file creates a database table called 'EMP' and creates a trigger on it
called LDAP_EMP which will synchronize all changes happening to the table with an
LDAP server. The changes to the database table are reflected/replicated to the LDAP
directory using the DBMS_LDAP package.
This script assumes the following:
LDAP server hostname: NULL (local host)
LDAP server portnumber: 389
Directory container for employee records: o=acme, dc=com
Username/Password for Directory Updates: cn=orcladmin/welcome
The aforementioned variables could be customized for different environments by
```

changing the appropriate variables in the code below.

Table Definition:

Employee Details(Columns) in Database Table(EMP):

EMP_ID	Number
FIRST_NAME	Varchar2
LAST_NAME	Varchar2
MANAGER_ID	Number
PHONE_NUMBER	Varchar2
MOBILE	Varchar2
ROOM_NUMBER	Varchar2
TITLE	Varchar2

LDAP Schema Definition & mapping to relational schema EMP:

Corresponding Data representation in LDAP directory:

DN	cn=FIRST_NAME LAST_NAME, o=acme, dc=com]
cn	FIRST_NAME LAST_NAME
sn	LAST_NAME
givenname	FIRST_NAME
manager	DN
telephonenumber	PHONE_NUMBER
mobile	MOBILE
employeeNumber	EMP_ID
userpassword	FIRST_NAME
objectclass	person
	organizationalperson
	inetOrgPerson
	top

MODIFIED (MM/DD/YY)

rbollu 07/21/00 - created

-Creating EMP table

PROMPT Dropping Table EMP ..

drop table EMP;

PROMPT Creating Table EMP ..

```
CREATE TABLE EMP (
    EMP_ID      NUMBER,           Employee Number
    FIRST_NAME  VARCHAR2(256),    First Name
    LAST_NAME   VARCHAR2(256),    Last Name
    MANAGER_ID  NUMBER,           Manager Number
    PHONE_NUMBER VARCHAR2(256),    Telephone Number
    MOBILE      VARCHAR2(256),    Mobile Number
    ROOM_NUMBER VARCHAR2(256),    Room Number
    TITLE       VARCHAR2(256)     Title in the company
);
```

```
--Creating Trigger LDAP_EMP

PROMPT Creating Trigger LDAP_EMP ..

CREATE OR REPLACE TRIGGER LDAP_EMP
AFTER INSERT OR DELETE OR UPDATE ON EMP
FOR EACH ROW

DECLARE
    retval    PLS_INTEGER;
    emp_session DBMS_LDAP.session;
    emp_dn     VARCHAR2(256);
    emp_rdn    VARCHAR2(256);
    emp_array  DBMS_LDAP.MOD_ARRAY;
    emp_vals   DBMS_LDAP.STRING_COLLECTION ;
    ldap_host  VARCHAR2(256);
    ldap_port  VARCHAR2(256);
    ldap_user  VARCHAR2(256);
    ldap_passwd VARCHAR2(256);
    ldap_base  VARCHAR2(256);
BEGIN

    retval      := -1;
    -- Customize the following variables as needed
    ldap_host   := NULL;
    ldap_port   := '389';
    ldap_user   := 'cn=orcladmin';
    ldap_passwd := 'welcome';
    ldap_base   := 'o=acme,dc=com';
    -- end of customizable settings

    DBMS_OUTPUT.PUT('Trigger [LDAP_EMP]: Replicating changes ');
    DBMS_OUTPUT.PUT_LINE('to directory .. ');
    DBMS_OUTPUT.PUT_LINE(RPAD('LDAP Host ',25,' ') || ': ' || ldap_host);
    DBMS_OUTPUT.PUT_LINE(RPAD('LDAP Port ',25,' ') || ': ' || ldap_port);

    -- Choosing exceptions to be raised by DBMS_LDAP library.
    DBMS_LDAP.USE_EXCEPTION := TRUE;

    -- Initialize ldap library and get session handle.
    emp_session := DBMS_LDAP.init(ldap_host,ldap_port);

    DBMS_OUTPUT.PUT_LINE (RPAD('Ldap session ',25,' ') || ': ' ||
        RAWTOHEX(SUBSTR(emp_session,1,8)) ||
        '(returned from init)');
```

```

-- Bind to the directory
retval := DBMS_LDAP.simple_bind_s(emp_session,
    ldap_user,ldap_passwd);

    DBMS_OUTPUT.PUT_LINE(RPAD('simple_bind_s Returns ',25,' ') || ' ': '
        || TO_CHAR(retval));

-- Process New Entry in the database

IF INSERTING THEN

    -- Create and setup attribute array for the New entry
    emp_array := DBMS_LDAP.create_mod_array(14);

    -- RDN to be - cn="FIRST_NAME LAST_NAME"

    emp_vals(1) := :new.FIRST_NAME || ' ' || :new.LAST_NAME;

    DBMS_LDAP.populate_mod_array(emp_array,DBMS_LDAP.MOD_ADD,
        'cn',emp_vals);

    emp_vals(1) := :new.LAST_NAME;

    DBMS_LDAP.populate_mod_array(emp_array,DBMS_LDAP.MOD_ADD,
        'sn',emp_vals);

    emp_vals(1) := :new.FIRST_NAME;

    DBMS_LDAP.populate_mod_array(emp_array,DBMS_LDAP.MOD_ADD,
        'givenname',emp_vals);

    emp_vals(1) := 'top';
    emp_vals(2) := 'person';
    emp_vals(3) := 'organizationalPerson';
    emp_vals(4) := 'inetOrgPerson';

    DBMS_LDAP.populate_mod_array(emp_array,DBMS_LDAP.MOD_ADD,
        'objectclass',emp_vals);

    emp_vals.DELETE;
    emp_vals(1) := :new.PHONE_NUMBER;

    DBMS_LDAP.populate_mod_array(emp_array,DBMS_LDAP.MOD_ADD,
        'telephonenumber',emp_vals);

    emp_vals(1) := :new.MOBILE;

```

```
DBMS_LDAP.populate_mod_array(emp_array,DBMS_LDAP.MOD_ADD,
                              'mobile',emp_vals);

emp_vals(1) := :new.ROOM_NUMBER;

DBMS_LDAP.populate_mod_array(emp_array,DBMS_LDAP.MOD_ADD,
                              'roomNumber',emp_vals);

emp_vals(1) := :new.TITLE;

DBMS_LDAP.populate_mod_array(emp_array,DBMS_LDAP.MOD_ADD,
                              'title',emp_vals);

emp_vals(1) := :new.EMP_ID;

DBMS_LDAP.populate_mod_array(emp_array,DBMS_LDAP.MOD_ADD,
                              'employeeNumber',emp_vals);

emp_vals(1) := :new.FIRST_NAME;

DBMS_LDAP.populate_mod_array(emp_array,DBMS_LDAP.MOD_ADD,
                              'userpassword',emp_vals);

-- DN for Entry to be Added under 'ldap_base' [o=acme, dc=com]

emp_dn := 'cn=' || :new.FIRST_NAME || ' ' ||
:new.LAST_NAME || ', ' || ldap_base ;
DBMS_OUTPUT.PUT_LINE(RPAD('Adding Entry for DN ',25,' ') || ': ['
|| emp_dn || ']');

-- Add new Entry to ldap directory
retval := DBMS_LDAP.add_s(emp_session,emp_dn,emp_array);
DBMS_OUTPUT.PUT_LINE(RPAD('add_s Returns ',25,' ') || ': '
|| TO_CHAR(retval));

-- Free attribute array (emp_array)
DBMS_LDAP.free_mod_array(emp_array);

END IF; -- INSERTING

-- Process Entry deletion in database

IF DELETING THEN

-- DN for Entry to be deleted under 'ldap_base' [o=acme, dc=com]

emp_dn := 'cn=' || :old.FIRST_NAME || ' ' ||
```

```

:old.LAST_NAME || ', ' || ldap_base ;
DBMS_OUTPUT.PUT_LINE(RPAD('Deleting Entry for DN ',25,' ') ||
    ': [' || emp_dn || ']');

-- Delete entry in ldap directory
retval := DBMS_LDAP.delete_s(emp_session,emp_dn);
    DBMS_OUTPUT.PUT_LINE(RPAD('delete_s Returns ',25,' ') || ': ' ||
        TO_CHAR(retval));

END IF; -- DELETING

-- Process updated Entry in database

IF UPDATING THEN

    -- Since two Table columns(in this case) constitute a RDN
    -- check for any changes and update RDN in ldap directory
    -- before updating any other attributes of the Entry.

    IF :old.FIRST_NAME <> :new.FIRST_NAME OR
        :old.LAST_NAME <> :new.LAST_NAME THEN

        emp_dn := 'cn=' || :old.FIRST_NAME || ' ' ||
            :old.LAST_NAME || ', ' || ldap_base;

        emp_rdn := 'cn=' || :new.FIRST_NAME || ' ' || :new.LAST_NAME;

        DBMS_OUTPUT.PUT_LINE(RPAD('Renaming OLD DN ',25,' ') ||
            ': [' || emp_dn || ']');
        DBMS_OUTPUT.PUT_LINE(RPAD(' => NEW RDN ',25,' ') ||
            ': [' || emp_rdn || ']');
        retval := DBMS_LDAP.modrdn2_s(emp_session,emp_dn,emp_rdn,
            DBMS_LDAP.MOD_DELETE);
        DBMS_OUTPUT.PUT_LINE(RPAD('modrdn2_s Returns ',25,' ') || ': ' ||
            TO_CHAR(retval));

    END IF;

    -- DN for Entry to be updated under 'ldap_base' [o=acme, dc=com]

    emp_dn := 'cn=' || :new.FIRST_NAME || ' ' ||
        :new.LAST_NAME || ', ' || ldap_base;

    DBMS_OUTPUT.PUT_LINE(RPAD('Updating Entry for DN ',25,' ') ||
        ': [' || emp_dn || ']');

    -- Create and setup attribute array(emp_array) for updated entry
    emp_array := DBMS_LDAP.create_mod_array(7);

```

```
emp_vals(1) := :new.LAST_NAME;

DBMS_LDAP.populate_mod_array(emp_array,DBMS_LDAP.MOD_REPLACE,
                             'sn',emp_vals);

emp_vals(1) := :new.FIRST_NAME;

DBMS_LDAP.populate_mod_array(emp_array,DBMS_LDAP.MOD_REPLACE,
                             'givenname',emp_vals);

emp_vals(1) := :new.PHONE_NUMBER;

DBMS_LDAP.populate_mod_array(emp_array,DBMS_LDAP.MOD_REPLACE,
                             'telephonenumber',emp_vals);

emp_vals(1) := :new.MOBILE;

DBMS_LDAP.populate_mod_array(emp_array,DBMS_LDAP.MOD_REPLACE,
                             'mobile',emp_vals);

emp_vals(1) := :new.ROOM_NUMBER;

DBMS_LDAP.populate_mod_array(emp_array,DBMS_LDAP.MOD_REPLACE,
                             'roomNumber',emp_vals);

emp_vals(1) := :new.TITLE;

DBMS_LDAP.populate_mod_array(emp_array,DBMS_LDAP.MOD_REPLACE,
                             'title',emp_vals);

emp_vals(1) := :new.EMP_ID;

DBMS_LDAP.populate_mod_array(emp_array,DBMS_LDAP.MOD_REPLACE,
                             'employeeNumber',emp_vals);

-- Modify entry in ldap directory
retval := DBMS_LDAP.modify_s(emp_session,emp_dn,emp_array);

        DBMS_OUTPUT.PUT_LINE(RPAD('modify_s Returns ',25,' ') || ': ' ||
                              TO_CHAR(retval));

-- Free attribute array (emp_array)
DBMS_LDAP.free_mod_array(emp_array);

END IF; -- UPDATING
```

```

-- Unbind from ldap directory
retval := DBMS_LDAP.unbind_s(emp_session);

DBMS_OUTPUT.PUT_LINE(RPAD('unbind_res Returns ',25,' ') || ' : ' ||
                      TO_CHAR(retval));

DBMS_OUTPUT.PUT_LINE('Directory operation Successful .. exiting');

-- Handle Exceptions
EXCEPTION
    WHEN OTHERS THEN
        -- TODO : should the trigger call unbind at this point ??
        -- what if the exception was raised from unbind itself ??

        DBMS_OUTPUT.PUT_LINE(' Error code      : ' || TO_CHAR(SQLCODE));
        DBMS_OUTPUT.PUT_LINE(' Error Message : ' || SQLERRM);
        DBMS_OUTPUT.PUT_LINE(' Exception encountered .. exiting');

END;
/
-----END OF trigger.sql-----

```

## PL/SQL API での検索方法

次の例は、DBMS\_LDAP API を使用して、PL/SQL プログラムの中で LDAP 検索を実行する方法を示したものです。この例では、前述のトリガーの例で作成したエントリを検索します。この例では、ベースが o=acme,dc=com であると前提し、サブツリー検索を実行して、そのベース・エントリに従属するエントリをすべて取り出します。次に示すコードは、`$ORACLE_HOME/ldap/demo/plsql` ディレクトリにある `search.sql` という名前のファイルに保存されています。

\$Header: \$

Copyright (c) Oracle Corporation 2000. All Rights Reserved.

FILE  
    search.sql

DESCRIPTION

This SQL file contains the PL/SQL code required to perform  
a typical search against an LDAP server.

This script assumes the following:  
LDAP server hostname: NULL (local host)  
LDAP server portnumber: 389

Directory container for employee records: o=acme, dc=com  
Username/Password for Directory Updates: cn=orcladmin/welcome

**NOTE**

Run this file after you have run the 'trigger.sql' and 'empdata.sql' scripts to see what entries were added by the database triggers.

MODIFIED (MM/DD/YY)  
akolli07/21/00 - created

set serveroutput on size 30000

```
DECLARE
    retval          PLS_INTEGER;
    my_session      DBMS_LDAP.session;
    my_attrs        DBMS_LDAP.string_collection;
    my_message      DBMS_LDAP.message;
    my_entry        DBMS_LDAP.message;
    entry_index     PLS_INTEGER;
    my_dn           VARCHAR2(256);
    my_attr_name    VARCHAR2(256);
    my_ber_elmt     DBMS_LDAP.ber_element;
    attr_index      PLS_INTEGER;
    i               PLS_INTEGER;
    my_vals         DBMS_LDAP.STRING_COLLECTION ;
    ldap_host       VARCHAR2(256);
    ldap_port       VARCHAR2(256);
    ldap_user       VARCHAR2(256);
    ldap_passwd     VARCHAR2(256);
    ldap_base       VARCHAR2(256);

BEGIN
    retval          := -1;

    -- Please customize the following variables as needed
    ldap_host       := NULL ;
    ldap_port       := '389';
    ldap_user       := 'cn=orcladmin';
    ldap_passwd     := 'welcome';
    ldap_base       := 'o=acme,dc=com';
    -- end of customizable settings

    DBMS_OUTPUT.PUT('DBMS_LDAP Search Example ');
    DBMS_OUTPUT.PUT_LINE('to directory .. ');
```

```

DBMS_OUTPUT.PUT_LINE(RPAD('LDAP Host ',25,' ') || ': ' || ldap_host);
DBMS_OUTPUT.PUT_LINE(RPAD('LDAP Port ',25,' ') || ': ' || ldap_port);

-- Choosing exceptions to be raised by DBMS_LDAP library.
DBMS_LDAP.USE_EXCEPTION := TRUE;

my_session := DBMS_LDAP.init(ldap_host,ldap_port);

DBMS_OUTPUT.PUT_LINE (RPAD('Ldap session ',25,' ') || ': ' ||
    RAWTOHEX(SUBSTR(my_session,1,8)) ||
    '(returned from init)');

-- bind to the directory
retval := DBMS_LDAP.simple_bind_s(my_session,
    ldap_user, ldap_passwd);

DBMS_OUTPUT.PUT_LINE(RPAD('simple_bind_s Returns ',25,' ') || ': ' ||
    TO_CHAR(retval));

-- issue the search
my_attrs(1) := '*'; -- retrieve all attributes
retval := DBMS_LDAP.search_s(my_session, ldap_base,
    DBMS_LDAP.SCOPE_SUBTREE,
    'objectclass=*',
    my_attrs,
    0,
    my_message);

DBMS_OUTPUT.PUT_LINE(RPAD('search_s Returns ',25,' ') || ': ' ||
    TO_CHAR(retval));
DBMS_OUTPUT.PUT_LINE (RPAD('LDAP message ',25,' ') || ': ' ||
    RAWTOHEX(SUBSTR(my_message,1,8)) ||
    '(returned from search_s)');

-- count the number of entries returned
retval := DBMS_LDAP.count_entries(my_session, my_message);
DBMS_OUTPUT.PUT_LINE(RPAD('Number of Entries ',25,' ') || ': ' ||
    TO_CHAR(retval));
DBMS_OUTPUT.PUT_LINE('-----');

-- get the first entry
my_entry := DBMS_LDAP.first_entry(my_session, my_message);
entry_index := 1;

```

```
-- Loop through each of the entries one by one
while my_entry IS NOT NULL loop
    -- print the current entry
    my_dn := DBMS_LDAP.get_dn(my_session, my_entry);
    -- DBMS_OUTPUT.PUT_LINE ('          entry #' || TO_CHAR(entry_index) ||
    -- ' entry ptr: ' || RAWTOHEX(SUBSTR(my_entry,1,8)));
    DBMS_OUTPUT.PUT_LINE ('          dn: ' || my_dn);
    my_attr_name := DBMS_LDAP.first_attribute(my_session,my_entry,
    my_ber_elmt);
    attr_index := 1;
    while my_attr_name IS NOT NULL loop
        my_vals := DBMS_LDAP.get_values (my_session, my_entry,
        my_attr_name);
        if my_vals.COUNT > 0 then
            FOR i in my_vals.FIRST..my_vals.LAST loop
                DBMS_OUTPUT.PUT_LINE('          ' || my_attr_name || ' : ' ||
                SUBSTR(my_vals(i),1,200));
            end loop;
        end if;
        my_attr_name := DBMS_LDAP.next_attribute(my_session,my_entry,
        my_ber_elmt);
        attr_index := attr_index+1;
    end loop;
    my_entry := DBMS_LDAP.next_entry(my_session, my_entry);
    DBMS_OUTPUT.PUT_LINE('=====');
    entry_index := entry_index+1;
end loop;

-- unbind from the directory
retval := DBMS_LDAP.unbind_s(my_session);
DBMS_OUTPUT.PUT_LINE(RPAD('unbind_res Returns ',25,' ') || ' : ' ||
    TO_CHAR(retval));

DBMS_OUTPUT.PUT_LINE('Directory operation Successful .. exiting');

-- Handle Exceptions
EXCEPTION
    WHEN OTHERS THEN
        DBMS_OUTPUT.PUT_LINE(' Error code      : ' || TO_CHAR(SQLCODE));
        DBMS_OUTPUT.PUT_LINE(' Error Message : ' || SQLERRM);
        DBMS_OUTPUT.PUT_LINE(' Exception encountered .. exiting');
END;
/
-----END OF trigger.sql-----
```

## PL/SQL LDAP API を使用したアプリケーションの作成

PL/SQL LDAP API を使用するには、まずデータベースにロードする必要があります。  
\$ORACLE\_HOME/rdbms/admin ディレクトリにある catldap.sql というスクリプトを使用してロードします。そのためには、SQL\*Plus コマンドライン・ツールを使用して、SYSDBA として接続する必要があります。

DBMS\_LDAP パッケージのロードに使用できるコマンド・シーケンスのサンプルを次に示します。

```
SQL> CONNECT / AS SYSDBA
SQL> @?/rdbms/admin/catldap.sql
```

## 依存性と制限事項

このリリースの PL/SQL LDAP API には、次の制限事項があります。

- マルチスレッド・サーバー (MTS)・モードのデータベースは使用できません。
- API から取得した LDAP セッション・ハンドルは、データベース・セッションの継続時間内のみ有効です。LDAP セッション・ハンドルを表に書き込んだり、他のデータベース・セッションで再利用することはできません。
- このリリースでは、同期型の LDAP API ファンクションのみサポートされています。
- PL/SQL LDAP API で作業するには、データベースへ接続する必要があります。クライアント側の PL/SQL エンジン (Oracle Forms など) ではデータベースへの接続が有効でないかぎり、この API を使用できません。

## PL/SQL のリファレンス

PL/SQL パッケージ DBMS\_LDAP には、PL/SQL プログラマが LDAP サーバーからデータへアクセスする際に使用できる関数とプロシージャが含まれています。この項では、すべての API ファンクションの詳細を説明します。前の各項を読んでから、この項を使用してください。

この項では、次の項目について説明します。

- [サブプログラムの概要](#)
- [例外の概要](#)
- [データ型の概要](#)
- [サブプログラム](#)

## サブプログラムの概要

表 3-1 DBMS\_LDAP API のサブプログラム

ファンクションまたはプロシージャ	説明
init ファンクション	init() ファンクションを使用すると、LDAP サーバーとのセッションが初期化されます。これにより、実際に LDAP サーバーとの接続が確立されます。
simple_bind_s ファンクション	simple_bind_s ファンクションを使用すると、ユーザー名およびパスワードに基づく、ディレクトリ・サーバーへの簡易認証を実行できます。
bind_s ファンクション	bind_s ファンクションを使用すると、ディレクトリ・サーバーへの複雑な認証を実行できます。
unbind_s ファンクション	unbind_s ファンクションは、アクティブな LDAP セッションのクローズに使用します。
compare_s ファンクション	compare_s ファンクションを使用すると、特定のエントリの特定の属性が特定の値を持っているかどうかをテストできます。
search_s ファンクション	search_s ファンクションを使用すると、LDAP サーバー内で同期検索が実行されます。これを実行すると、サーバーからすべての検索結果が送信されるか、検索要求がサーバーによってタイムアウトになるまで、PL/SQL 環境に制御が戻されません。
search_st ファンクション	search_st ファンクションを使用すると、LDAP サーバー内で、クライアント側のタイムアウトを使用して同期検索が実行されます。これを実行すると、サーバーからすべての検索結果が送信されるか、検索要求がクライアントまたはサーバーによってタイムアウトになるまで、PL/SQL 環境に制御が戻されません。
first_entry ファンクション	first_entry ファンクションを使用すると、search_s または search_st で戻された結果セット内の最初のエントリを取り出せます。
next_entry ファンクション	next_entry() ファンクションを使用すると、検索操作の結果セット内の次のエントリを取り出せます。
count_entries ファンクション	このファンクションは、結果セット内のエントリ数のカウントに使用します。また、first_entry() ファンクションおよび next_entry() ファンクションと組み合わせて使用すると、結果セットの全探索時に残っているエントリの数をカウントすることもできます。

表 3-1 DBMS\_LDAP API のサブプログラム (続き)

ファンクションまたはプロシージャ	説明
<code>first_attribute</code> ファンクション	<code>first_attribute()</code> ファンクションを使用すると、結果セットの中から、指定したエントリの最初の属性がフェッチされます。
<code>next_attribute</code> ファンクション	<code>next_attribute()</code> ファンクションを使用すると、結果セットの中から、指定したエントリの次の属性がフェッチされます。
<code>get_dn</code> ファンクション	<code>get_dn()</code> ファンクションを使用すると、結果セットの中から、指定したエントリの X.500 識別名が取り出されます。
<code>get_values</code> ファンクション	<code>get_values()</code> ファンクションを使用すると、特定のエントリの特定の属性に関連する値をすべて取り出せます。
<code>get_values_len</code> ファンクション	<code>get_values_len()</code> ファンクションを使用すると、バイナリ構文を持つ属性の値を取り出せます。
<code>delete_s</code> ファンクション	<code>delete_s</code> ファンクションを使用すると、LDAP ディレクトリ情報ツリー内のリーフ・エントリを削除できます。
<code>modrdn2_s</code> ファンクション	<code>modrdn2_s()</code> ファンクションを使用すると、エントリの相対識別名を変更できます。
<code>err2string</code> ファンクション	<code>err2string()</code> ファンクションを使用すると、LDAP エラー・コードを、API の動作環境で使用されている各国語の文字列に変換できます。
<code>create_mod_array</code> ファンクション	<code>create_mod_array()</code> ファンクションを使用すると、 <code>modify_s()</code> ファンクションを使用してエントリに適用される変更配列に、メモリーが割り当てられます。
<code>populate_mod_array</code> プロシージャ (文字列バージョン)	追加操作または変更操作に、1 組の属性情報を移入します。このプロシージャ・コールは、 <code>DBMS_LDAP.create_mod_array()</code> をコールした後に行う必要があります。
<code>populate_mod_array</code> プロシージャ (バイナリ・バージョン)	追加操作または変更操作に、1 組の属性情報を移入します。このプロシージャ・コールは、 <code>DBMS_LDAP.create_mod_array()</code> をコールした後に行う必要があります。
<code>modify_s</code> ファンクション	既存の LDAP ディレクトリ・エントリの同期変更を実行します。 <code>add_s</code> をコールする前に、 <code>DBMS_LDAP.create_mod_array()</code> と <code>DBMS_LDAP.populate_mod_array()</code> をコールする必要があります。
<code>add_s</code> ファンクション	LDAP ディレクトリに新規エントリを同期的に追加します。 <code>add_s</code> をコールする前に、 <code>DBMS_LDAP.create_mod_array()</code> と <code>DBMS_LDAP.populate_mod_array()</code> をコールする必要があります。

表 3-1 DBMS\_LDAP API のサブプログラム (続き)

ファンクションまたはプロシージャ	説明
<code>free_mod_array</code> プロシージャ	<code>DBMS_LDAP.create_mod_array()</code> によって割り当てられたメモリーを解放します。
<code>count_values</code> ファンクション	<code>DBMS_LDAP.get_values()</code> によって戻された値の数をカウントします。
<code>count_values_len</code> ファンクション	<code>DBMS_LDAP.get_values_len()</code> によって戻された値の数をカウントします。
<code>rename_s</code> ファンクション	LDAP エントリの名前を同期的に変更します。
<code>explode_dn</code> ファンクション	識別名 (DN) を個々の構成要素に分割します。
<code>open_ssl</code> ファンクション	すでに確立されている LDAP 接続を介して SSL (Secure Sockets Layer) 接続を確立します。

例外の概要

RDBMS リリース 8.1.7 とともに出荷される DBMS\_LDAP を使用すると、次の例外が発生する場合があります。

表 3-2 DBMS\_LDAP 例外の概要

例外名	Oracle エラー番号	例外の原因
<code>general_error</code>	31202	関係する特定の PL/SQL 例外がないエラーが発生すると常に呼び出されます。エラー文字列には、ユーザーが使用している各国語で問題が説明されます。
<code>init_failed</code>	31203	<code>DBMS_LDAP.init()</code> でなんらかの問題がある場合に呼び出されます。
<code>invalid_session</code>	31204	DBMS_LDAP パッケージのファンクションおよびプロシージャに無効なセッション・ハンドルが渡された場合に呼び出されます。
<code>invalid_auth_method</code>	31205	<code>DBMS_LDAP.bind_s()</code> で要求された認証方法がサポートされていない場合に呼び出されます。
<code>invalid_search_scope</code>	31206	検索の範囲が無効な場合に、検索ファンクションのいずれかによって呼び出されます。
<code>invalid_search_time_val</code>	31207	時間に基づく検索ファンクション <code>DBMS_LDAP.search_st()</code> の制限時間に無効な値を指定した場合に呼び出されます。

表 3-2 DBMS\_LDAP 例外の概要 (続き)

例外名	Oracle エラー番号	例外の原因
invalid_message	31208	検索操作によるエントリの取得を結果セット全体にわたって反復するファンクションに、無効なメッセージ・ハンドルが指定された場合に呼び出されます。
count_entry_error	31209	DBMS_LDAP.count_entries で指定した結果セット内のエントリをカウントできない場合に呼び出されます。
get_dn_error	31210	DBMS_LDAP.get_dn によって取り出されるエントリの識別名 (DN) が NULL である場合に呼び出されます。
invalid_entry_dn	31211	エントリを変更、追加または改名するファンクションに無効なエントリ識別名 (DN) を指定した場合に呼び出されます。
invalid_mod_array	31212	変更配列を引数として取る関数に、無効な変更配列を指定した場合に呼び出されます。
invalid_mod_option	31213	DBMS_LDAP.populate_mod_array で指定した変更オプションが、MOD_ADD、MOD_DELETE または MOD_REPLACE ではなかった場合に呼び出されます。
invalid_mod_type	31214	DBMS_LDAP.populate_mod_array によって変更される属性の型が NULL である場合に呼び出されます。
invalid_mod_value	31215	DBMS_LDAP.populate_mod_array で指定した属性を NULL 値で更新しようとした場合に呼び出されます。
invalid_rdn	31216	相対識別名 (RDN) の値が NULL である場合に、有効な RDN を想定するファンクションおよびプロシージャによって呼び出されます。
invalid_newparent	31217	DBMS_LDAP.rename_s によって改名されるエントリの新しい親が NULL である場合に呼び出されます。
invalid_deleteoldrdn	31218	DBMS_LDAP.rename_s の deleteoldrdn パラメータが無効な場合に呼び出されます。
invalid_notypes	31219	DBMS_LDAP.explode_dn の notypes パラメータが無効な場合に呼び出されます。

表 3-2 DBMS\_LDAP 例外の概要（続き）

例外名	Oracle エラー番号	例外の原因
invalid_ssl_wallet_loc	31220	Wallet の場所が NULL であるが SSL 認証モードが有効な Wallet を必要とする場合に、DBMS_LDAP.open_ssl によって呼び出されます。
invalid_ssl_wallet_password	31221	DBMS_LDAP.open_ssl で指定した Wallet パスワードが NULL である場合に呼び出されます。
invalid_ssl_auth_mode	31222	SSL 認証モードが 1、2 または 3 ではない場合に DBMS_LDAP.open_ssl によって呼び出されます。
mts_mode_not_supported	31398	init()、bind_s() または simple_bind_s() ファンクションが MTS モードでコールされた場合に呼び出されます。

データ型の概要

DBMS\_LDAP パッケージでは、次のデータ型を使用します。

表 3-3 DBMS\_LDAP データ型の概要

データ型	用途
SESSION	LDAP セッションのハンドルを保持するために使用します。API のほとんどすべてのファンクションでは、作業のために、有効な LDAP セッションが必要となります。
MESSAGE	結果セットから取り出されたメッセージのハンドルを保持するために使用します。このデータ型は、エントリの属性および値を処理するすべてのファンクションで使用します。
MOD_ARRAY	modify_s() または add_s() に渡される変更配列のハンドルを保持するために使用します。
TIMEVAL	制限時間を必要とする LDAP API ファンクションに制限時間の情報を渡すために使用します。
BER_ELEMENT	受信メッセージのデコードに使用される BER 構造体のハンドルを保持するために使用します。
STRING_COLLECTION	LDAP サーバーに渡すことができる VARCHAR2 文字列のリストを保持するために使用します。
BINVAL_COLLECTION	バイナリ・データを表す RAW データのリストを保持するために使用します。
BERVAL_COLLECTION	変更配列の代入に使用される BEREVAL 値のリストを保持するために使用します。

## サブプログラム

### init ファンクション

init() ファンクションを使用すると、LDAP サーバーとのセッションが初期化されます。これにより、実際に LDAP サーバーとの接続が確立されます。

#### 構文

```
FUNCTION init
(
    hostname IN VARCHAR2,
    portnum  IN PLS_INTEGER
)
    RETURN SESSION;
```

#### パラメータ

表 3-4 INIT ファンクションのパラメータ

パラメータ	説明
hostname	スペースで区切られたホスト名のリスト、または LDAP サーバーを実行している接続先のホストの IP アドレスを示す点区切りの文字列が入ります。リスト内の各ホスト名には、ポート番号を含めることもできます。ポート番号とホスト自体はコロン (:) で区切ります。ホストへの接続はリストの順序に従って試みられ、最初にホストへの接続が成功した時点で終了します。
portnum	接続先の TCP ポート番号が入ります。ホストにポート番号を含めた場合は、このパラメータは無視されます。このパラメータを指定せず、ホスト名にもポート番号を含めていない場合は、デフォルトのポート番号 389 が指定されたものと見なされます。

#### 戻り値

表 3-5 INIT ファンクションの戻り値

値	説明
SESSION (ファンクション戻り値)	以後の API のコールに使用できる LDAP セッション・ハンドルです。

例外

表 3-6 INIT ファンクションの例外

例外	説明
init_failed	LDAP サーバーとの通信中に問題が発生した場合に呼び出されます。
mts_mode_not_supported	MTS サービスを使用してデータベースにログインしているユーザー・セッションから DBMS_LDAP.init() がコールされた場合に呼び出されます。
general_error	その他すべてのエラーに対応します。この例外に関連するエラー文字列で、エラーの詳細が説明されます。

使用方法

DBMS\_LDAP.init() は、LDAP サーバーとのセッションを確立するために最初にコールする必要があるファンクションです。DBMS\_LDAP.init() ファンクションの戻り値はセッション・ハンドルです。セッション・ハンドルとは、セッションに関連する後続のコールへ渡す必要がある不透過的構造体へのポインタです。このルーチンでセッションを初期化できない場合は NULL が戻され、INIT\_FAILED 例外が呼び出されます。init() をコールした後、DBMS\_LDAP.bind\_s または DBMS\_LDAP.simple\_bind\_s() を使用して認証を行う必要があります。

関連項目

DBMS\_LDAP.simple\_bind\_s()、DBMS\_LDAP.bind\_s()

## simple\_bind\_s ファンクション

simple\_bind\_s ファンクションを使用すると、ユーザー名およびパスワードに基づく、ディレクトリ・サーバーへの簡易認証を実行できます。

### 構文

```
FUNCTION simple_bind_s
(
    ld      IN SESSION,
    dn      IN VARCHAR2,
    passwd  IN VARCHAR2
)
RETURN PLS_INTEGER;
```

### パラメータ

表 3-7 SIMPLE\_BIND\_S ファンクションのパラメータ

パラメータ	説明
ld	有効な LDAP セッション・ハンドルです。
dn	ログイン時に使用するユーザー識別名です。
passwd	パスワードを含む文字列です。

### 戻り値

表 3-8 SIMPLE\_BIND\_S ファンクションの戻り値

値	説明
PLS_INTEGER (ファンクション戻り値)	正常終了した場合の戻り値は DBMS_LDAP.SUCCESS です。問題があった場合は、次の例外のいずれかが呼び出されます。

### 例外

表 3-9 SIMPLE\_BIND\_S ファンクションの例外

例外	説明
invalid_session	セッション・ハンドル ld が無効な場合に呼び出されます。
mts_mode_not_supported	MTS サービスでログインしているユーザー・セッションから DBMS_LDAP.init() がコールされた場合に呼び出されます。
general_error	その他すべてのエラーに対応します。この例外に関連するエラー文字列で、エラーの詳細が説明されます。

### **使用方法**

DBMS\_LDAP.simple\_bind\_s() を使用すると、ディレクトリ識別名およびディレクトリ・パスワードがすでに分かっているユーザーを認証できます。DBMS\_LDAP.init() のコールで有効な LDAP セッション・ハンドルを取得してから、この関数をコールしてください。

## bind\_s ファンクション

bind\_s ファンクションを使用すると、ディレクトリ・サーバーへの高度な認証を実行できます。

### 構文

```
FUNCTION bind_s
(
    ld      IN SESSION,
    dn      IN VARCHAR2,
    cred IN VARCHAR2,
    meth IN PLS_INTEGER
)
RETURN PLS_INTEGER;
```

### パラメータ

**表 3-10 BIND\_S ファンクションのパラメータ**

パラメータ	説明
ld	有効な LDAP セッション・ハンドルです。
dn	ログイン時に使用するユーザー識別名です。
cred	認証に使用する資格証明を含んだ文字列です。
meth	認証方法です。

### 戻り値

**表 3-11 BIND\_S ファンクションの戻り値**

値	説明
PLS_INTEGER (ファンクション戻り値)	正常終了した場合の戻り値は DBMS_LDAP.SUCCESS です。問題があった場合は、次の例外が呼び出されます。

例外

表 3-12 BIND\_S ファンクションの例外

例外	説明
invalid_session	セッション・ハンドルldが無効な場合に呼び出されます。
invalid_auth_method	要求した認証方法がサポートされていない場合に呼び出されます。
mts_mode_not_supported	MTS サービスにログインしているユーザー・セッションからコールされた場合に呼び出されます。
general_error	その他すべてのエラーに対応します。この例外に関連するエラー文字列で、エラーの詳細が説明されます。

使用方法

DBMS\_LDAP.bind\_s() を使用すると、ユーザーを認証できます。DBMS\_LDAP.init() のコールで有効な LDAP セッション・ハンドルを取得してから、この関数をコールしてください。

関連項目

DBMS\_LDAP.init()、DBMS\_LDAP.simple\_bind\_s()

## unbind\_s ファンクション

unbind\_s ファンクションは、アクティブな LDAP セッションのクローズに使用します。

### 構文

```
FUNCTION unbind_s  
(  
    ld IN SESSION  
)  
    RETURN PLS_INTEGER;
```

### パラメータ

表 3-13 UNBIND\_S ファンクションのパラメータ

パラメータ	説明
ld	有効な LDAP セッション・ハンドルです。

### 戻り値

表 3-14 UNBIND\_S ファンクションの戻り値

値	説明
PLS_INTEGER (ファンクション戻り値)	正常終了した場合の戻り値は DBMS_LDAP.SUCCESS です。それ以外の場合は、次の例外のいずれかが呼び出されます。

### 例外

表 3-15 UNBIND\_S ファンクションの例外

例外	説明
invalid_session	セッション・ハンドル ld が無効な場合に呼び出されます。
general_error	その他すべてのエラーに対応します。この例外に関連するエラー文字列で、エラーの詳細が説明されます。

### 使用方法

unbind\_s() ファンクションを使用すると、サーバーにバインド解除要求が送信され、LDAP セッションに関連するオープンな接続がすべてクローズされ、セッション・ハンドルに関連するリソースがすべて処理された後に値が戻されます。この関数をコールすると、セッション・ハンドル ld が無効になるため、これ以降に、ld を使用して LDAP API コールを行うことはできません。

**関連項目**

DBMS\_LDAP.bind\_s()、DBMS\_LDAP.simple\_bind\_s()

## compare\_s ファンクション

compare\_s ファンクションを使用すると、特定のエントリの特定の属性が特定の値を持っているかどうかをテストできます。

### 構文

```
FUNCTION compare_s
(
    ld      IN SESSION,
    dn      IN VARCHAR2,
    attr    IN VARCHAR2,
    value   IN VARCHAR2
)
RETURN PLS_INTEGER;
```

### パラメータ

**表 3-16 COMPARE\_S ファンクションのパラメータ**

パラメータ	説明
ld	有効な LDAP セッション・ハンドルです。
dn	比較の対象となるエントリの名前です。
attr	比較の対象となる属性です。
value	比較の対象となる文字列の属性値です。

### 戻り値

**表 3-17 COMPARE\_S ファンクションの戻り値**

値	説明
PLS_INTEGER (ファンクション戻り値)	属性の値が指定した値と一致した場合の戻り値は COMPARE_TRUE です。  属性の値が指定した値と一致しない場合の戻り値は COMPARE_FALSE です。

例外

表 3-18 COMPARE\_S ファンクションの例外

例外	説明
invalid_session	セッション・ハンドル ld が無効な場合に呼び出されます。
general_error	その他すべてのエラーに対応します。この例外に関連するエラー文字列で、エラーの詳細が説明されます。

使用方法

compare\_s ファンクションを使用すると、ディレクトリ・サーバーに格納されている特定の属性の値が、特定の値と一致するかどうかを確認できます。この操作は、比較が可能な構文定義を持つ属性についてのみ実行できます。compare\_s ファンクションは、init() ファンクションで有効な LDAP セッション・ハンドルを取得し、bind\_s() ファンクションまたは simple\_bind\_s() ファンクションを使用してこのセッション・ハンドルを認証してから、コールしてください。

関連項目

DBMS\_LDAP.bind\_s()

## search\_s ファンクション

search\_s ファンクションを使用すると、LDAP サーバー内で同期検索が実行されます。これを実行すると、サーバーからすべての検索結果が送信されるか、検索要求がサーバーによってタイムアウトになるまでは、PL/SQL 環境に制御が戻されません。

### 構文

```
FUNCTION search_s
(
    ld          IN  SESSION,
    base        IN  VARCHAR2,
    scope       IN  PLS_INTEGER,
    filter      IN  VARCHAR2,
    attrs       IN  STRING_COLLECTION,
    attronly    IN  PLS_INTEGER,
    res         OUT MESSAGE
)
RETURN PLS_INTEGER;
```

### パラメータ

**表 3-19 SEARCH\_S ファンクションのパラメータ**

パラメータ	説明
ld	有効な LDAP セッション・ハンドルです。
base	検索の開始点となるエントリの識別名 (DN) です。
scope	SCOPE_BASE (0x00)、SCOPE_ONELEVEL (0x01) または SCOPE_SUBTREE (0x02) のいずれかで、検索範囲を指定します。
filter	検索フィルタを表す文字列です。この値を NULL にすると、すべてのエントリに一致するフィルタ (objectclass=*) を使用するように指定できます。
attrs	一致した各エントリのどの属性を戻すかを指定する文字列の集合です。このパラメータを NULL にすると、取得可能なユーザー属性がすべて取り出されます。文字列 NO_ATTRS ("1.1") を配列内の唯一の文字列として使用すると、サーバーから属性型を戻さないように指定できます。attrs 配列の中で、文字列 ALL_USER_ATTRS ("*") をなんらかの操作属性名とともに使用すると、すべてのユーザー属性に加えて、リストした操作属性を戻すように指定できます。

表 3-19 SEARCH\_S ファンクションのパラメータ（続き）

パラメータ	説明
attrsonly	属性の型と値の両方を戻す場合には 0（ゼロ）、属性の型のみを要求する場合には 0（ゼロ）以外を指定する必要があるブール値です。
res	コールの終了時に検索結果が入る結果パラメータです。戻される結果がない場合、*res は NULL に設定されます。

戻り値

表 3-20 SEARCH\_S ファンクションの戻り値

値	説明
PLS_INTEGER（ファンクション戻り値）	検索操作が正常終了した場合の戻り値は DBMS_LDAP.SUCCESS です。その他の場合は例外が呼び出されます。
res（OUT パラメータ）	検索が正常終了してエントリがあった場合、このパラメータは NON-NULL 値に設定されます。この値を使用すると、結果セットからエントリを取り出すことができます。

例外

表 3-21 SEARCH\_S ファンクションの例外

例外	説明
invalid_session	セッション・ハンドル ld が無効な場合に呼び出されます。
invalid_search_scope	検索範囲が、SCOPE_BASE、SCOPE_ONELEVEL または SCOPE_SUBTREE ではない場合に呼び出されます。
general_error	その他すべてのエラーに対応します。この例外に関連するエラー文字列で、エラーの詳細が説明されます。

使用方法

search\_s() ファンクションを使用すると検索操作が発行されます。検索操作が発行されると、サーバーからすべての検索結果が戻されるまでは、ユーザー環境に制御が戻されません。検索によって戻されるエントリがある場合、res パラメータの中に収められます。このパラメータはコール元に対しては不透過です。エントリ、属性、値などは、後述の解析ルーチンをコールすることで抽出できます。

関連項目

DBMS\_LDAP.search\_st()、DBMS\_LDAP.first\_entry()、DBMS\_LDAP.next\_entry

## search\_st ファンクション

search\_st ファンクションを使用すると、LDAP サーバー内で、クライアント側のタイムアウトを使用して同期検索が実行されます。これを実行すると、サーバーからすべての検索結果が送信されるか、検索要求がクライアントまたはサーバーによってタイムアウトになるまでは、PL/SQL 環境に制御が戻されません。

### 構文

```
FUNCTION search_st
(
    ld          IN  SESSION,
    base        IN  VARCHAR2,
    scope       IN  PLS_INTEGER,
    filter       IN  VARCHAR2,
    attrs       IN  STRING_COLLECTION,
    attronly    IN  PLS_INTEGER,
    tv          IN  TIMEVAL,
    res         OUT MESSAGE
)
RETURN PLS_INTEGER;
```

### パラメータ

**表 3-22 SEARCH\_ST ファンクションのパラメータ**

パラメータ	説明
ld	有効な LDAP セッション・ハンドルです。
base	検索の開始点となるエントリの識別名 (DN) です。
scope	SCOPE_BASE (0x00)、SCOPE_ONELEVEL (0x01) または SCOPE_SUBTREE (0x02) のいずれかで、検索範囲を指定します。
filter	検索フィルタを表す文字列です。この値を NULL にすると、すべてのエントリに一致するフィルタ (objectclass=*) を使用するよう指定できます。
attrs	一致した各エントリのどの属性を戻すかを指定する文字列の集合です。このパラメータを NULL にすると、取得可能なユーザー属性がすべて取り出されます。文字列 NO_ATTRS ("1.1") を配列内の唯一の文字列として使用すると、サーバーから属性型を戻さないように指定できます。attrs 配列の中で、文字列 ALL_USER_ATTRS ("*") をなんらかの操作属性名とともに使用すると、すべてのユーザー属性に加えて、リストした操作属性を戻すように指定できます。

表 3-22 SEARCH\_ST ファンクションのパラメータ（続き）

パラメータ	説明
attrsonly	属性の型と値の両方を戻す場合には 0（ゼロ）、属性の型のみを要求する場合には 0（ゼロ）以外を指定する必要があるブール値です。
tv	この検索で使用する必要があるタイムアウト値です（秒およびミリ秒単位で表します）。
res	コールの終了時に検索結果が入る結果パラメータです。戻される結果がない場合、*res は NULL に設定されます。

戻り値

表 3-23 SEARCH\_ST ファンクションの戻り値

値	説明
PLS_INTEGER（ファンクション戻り値）	検索操作が正常終了した場合の戻り値は DBMS_LDAP.SUCCESS です。その他の場合は例外が呼び出されます。
res（OUT パラメータ）	検索が正常終了してエントリがあった場合、このパラメータは NON_NULL 値に設定されます。この値を使用すると、結果セットからエントリを取り出すことができます。

例外

表 3-24 SEARCH\_ST ファンクションの例外

例外	説明
invalid_session	セッション・ハンドル ld が無効な場合に呼び出されます。
invalid_search_scope	検索範囲が、SCOPE_BASE、SCOPE_ONELEVEL または SCOPE_SUBTREE ではない場合に呼び出されます。
invalid_search_time_value	タイムアウトに指定した時間の値が無効な場合に呼び出されます。
general_error	その他すべてのエラーに対応します。この例外に関連するエラー文字列で、エラーの詳細が説明されます。

使用方法

このファンクションは DBMS\_LDAP.search\_s() に類似していますが、タイムアウト値を指定する必要があるという点に違いがあります。

関連項目

DBMS\_LDAP.search\_s()、DBML\_LDAP.first\_entry()、DBMS\_LDAP.next\_entry

## first\_entry ファンクション

first\_entry ファンクションを使用すると、search\_s() または search\_st() で戻された結果セット内の最初のエントリを取り出せます。

### 構文

```
FUNCTION first_entry
(
    ld IN SESSION,
    msg IN MESSAGE
)
RETURN MESSAGE;
```

### パラメータ

表 3-25 FIRST\_ENTRY ファンクションのパラメータ

パラメータ	説明
ld	有効な LDAP セッション・ハンドルです。
msg	検索結果です。同期検索ルーチンのいずれかをコールして取得されるものと同一です。

### 戻り値

表 3-26 FIRST\_ENTRY の戻り値

値	説明
MESSAGE (ファンクション戻り値)	LDAP サーバーから戻されたエントリのリスト中の最初のエントリのハンドルです。エラーがあった場合は NULL に設定され、例外が呼び出されます。

### 例外

表 3-27 FIRST\_ENTRY の例外

例外	説明
invalid_session	セッション・ハンドル ld が無効な場合に呼び出されます。
invalid_message	受信した msg ハンドルが無効な場合に呼び出されます。

### 使用方法

first\_entry() ファンクションは、検索操作からの結果を取り出すために必ず最初にコールする必要がある関数です。

**関連項目**

DBMS\_LDAP.next\_entry()、DBMS\_LDAP.search\_s()、DBMS\_LDAP.search\_st()

## next\_entry ファンクション

next\_entry() ファンクションを使用すると、検索操作による結果セット内の次のエントリを取り出すことができます。

### 構文

```
FUNCTION next_entry  
(  
    ld IN SESSION,  
    msg IN MESSAGE  
)  
    RETURN MESSAGE;
```

### パラメータ

表 3-28 NEXT\_ENTRY ファンクションのパラメータ

パラメータ	説明
ld	有効な LDAP セッション・ハンドルです。
msg	検索結果です。同期検索ルーチンのいずれかをコールして取得されるものと同一です。

### 戻り値

表 3-29 NEXT\_ENTRY ファンクションの戻り値

値	説明
MESSAGE	LDAP サーバーから戻されたエントリのリスト中の次のエントリのハンドルです。エラーがあった場合は NULL に設定されて、例外が呼び出されます。

### 例外

表 3-30 NEXT\_ENTRY ファンクションの例外

例外	説明
invalid_session	セッション・ハンドル ld が無効な場合に呼び出されます。
invalid_message	受信した msg ハンドルが無効な場合に呼び出されます。

### 使用方法

`next_entry()` ファンクションは、必ず `first_entry()` ファンクションの後にコールする必要があります。また、リスト中の次のエントリをフェッチするには、正常終了した `next_entry()` のコールの戻り値を、次の `next_entry()` のコールで `msg` 引数として使用する必要があります。

### 関連項目

`DBMS_LDAP.first_entry()`、`DBMS_LDAP.search_s()`、`DBMS_LDAP.search_st()`

## count\_entries ファンクション

このファンクションは、結果セット内のエントリ数のカウントに使用します。また、`first_entry()` ファンクションおよび `next_entry()` ファンクションと組み合わせて使用すると、結果セットの全探索時に残っているエントリ の数をカウントすることもできます。

### 構文

```
FUNCTION count_entries
(
    ld IN SESSION,
    msg IN MESSAGE
)
RETURN PLS_INTEGER;
```

### パラメータ

表 3-31 COUNT\_ENTRY ファンクションの戻り値

パラメータ	説明
ld	有効な LDAP セッション・ハンドルです。
msg	検索結果です。同期検索ルーチンのいずれかをコールして取得されるものと同一です。

### 戻り値

表 3-32 COUNT\_ENTRY ファンクションの戻り値

値	説明
PLS_INTEGER (ファンクション戻り値)	結果セット中にエントリがある場合の戻り値は 0 (ゼロ) 以外です。 問題があった場合の戻り値は -1 です。

### 例外

表 3-33 COUNT\_ENTRY ファンクションの例外

例外	説明
invalid_session	セッション・ハンドル ld が無効な場合に呼び出されます。
invalid_message	受信した msg ハンドルが無効な場合に呼び出されます。
count_entry_error	エントリのカウント中に問題があった場合に呼び出されます。

### 使用方法

`count_entries()` の戻り値は、結果セットに含まれるエントリの数です。`res` パラメータが無効な場合など、なんらかのエラーが発生した場合は、-1 が戻されます。`first_message()`、`next_message()`、`first_entry()`、`next_entry()`、`first_reference()`、`next_reference()` によって戻されたメッセージ、エントリまたは参照について `count_entries()` をコールする場合は、結果セットの残りのエントリ数をカウントすることもできます。

### 関連項目

`DBMS_LDAP.first_entry()`、`DBMS_LDAP.next_entry()`

## first\_attribute ファンクション

first\_attribute() ファンクションを使用すると、結果セットの中から、指定したエントリの最初の属性がフェッチされます。

### 構文

```
FUNCTION first_attribute
(
    ld          IN  SESSION,
    msg         IN  MESSAGE,
    ber_elem OUT BER_ELEMENT
)
RETURN VARCHAR2;
```

### パラメータ

表 3-34 FIRST\_ATTRIBUTE ファンクションのパラメータ

パラメータ	説明
ld	有効な LDAP セッション・ハンドルです。
msg	属性を調べる対象となるエントリです。first_entry() または next_entry() の戻り値と同一です。
ber_elem	エントリ内のどの属性が読み取られたのかを記録するために使用される BER ELEMENT のハンドルです。

### 戻り値

表 3-35 FIRST\_ATTRIBUTE ファンクションの戻り値

値	説明
VARCHAR2 (ファンクション戻り値)	属性が存在する場合の戻り値は、その属性の名前です。 属性が存在しない場合、またはエラーが発生した場合の戻り値は NULL です。
ber_elem	DBMS_LDAP.next_attribute() で、すべての属性に対して同一処理を反復するために使用するハンドルです。

例外

表 3-36 FIRST\_ATTRIBUTE ファンクションの例外

例外	説明
invalid_session	セッション・ハンドルldが無効な場合に呼び出されます。
invalid_message	受信したmsgハンドルが無効な場合に呼び出されます。

使用方法

エントリの様々な属性に対して属性名の取出しを繰り返し行うには、first\_attribute() のパラメータとして戻された BER\_ELEMENT のハンドルを、次の next\_attribute() のコールで使用する必要があります。また、first\_attribute() のコールで戻された属性の名前を、get\_values() または get\_values\_len() のコールで使用すると、その属性の値を取得できます。

関連項目

DBMS\_LDAP.next\_attribute(), DBMS\_LDAP.get\_values(), DBMS\_LDAP.get\_values\_len(), DBMS\_LDAP.first\_entry(), DBMS\_LDAP.next\_entry()

## next\_attribute ファンクション

next\_attribute() ファンクションを使用すると、結果セットの中から、指定したエントリの次の属性がフェッチされます。

### 構文

```
FUNCTION next_attribute  
(  
    ld          IN SESSION,  
    msg         IN MESSAGE,  
    ber_elem IN BER_ELEMENT  
)  
    RETURN VARCHAR2;
```

### パラメータ

表 3-37 NEXT\_ATTRIBUTE ファンクションのパラメータ

パラメータ	説明
ld	有効な LDAP セッション・ハンドルです。
msg	属性を調べる対象となるエントリです。first_entry() または next_entry() の戻り値と同一です。
ber_elem	エントリ内のどの属性が読み取られたのかを記録するために使用される BER ELEMENT のハンドルです。

### 戻り値

表 3-38 NEXT\_ATTRIBUTE ファンクションの戻り値

値	説明
VARCHAR2 (ファンクション戻り値)	属性が存在する場合の戻り値はその属性の名前です。

### 例外

表 3-39 NEXT\_ATTRIBUTE ファンクションの例外

例外	説明
invalid_session	セッション・ハンドル ld が無効な場合に呼び出されます。
invalid_message	受信した msg ハンドルが無効な場合に呼び出されます。

### 使用方法

エントリの様々な属性に対して属性名の取出しを繰り返し行うには、`first_attribute()` のパラメータとして戻された `BER_ELEMENT` のハンドルを、次の `next_attribute()` のコールで使用する必要があります。また、`next_attribute()` のコールで戻された属性の名前を、`get_values()` または `get_values_len()` のコールで使用すると、その属性の値を取得できます。

### 関連項目

`DBMS_LDAP.first_attribute()`、`DBMS_LDAP.get_values()`、`DBMS_LDAP.get_values_len()`、`DBMS_LDAP.first_entry()`、`DBMS_LDAP.next_entry()`

get\_dn ファンクション

get\_dn() ファンクションを使用すると、結果セットの中から、指定したエントリの X.500 識別名が取り出されます。

構文

```
FUNCTION get_dn
(
    ld IN SESSION,
    msg IN MESSAGE
)
RETURN VARCHAR2;
```

パラメータ

表 3-40 GET\_DN ファンクションのパラメータ

パラメータ	説明
ld	有効な LDAP セッション・ハンドルです。
msg	識別名 (DN) が戻り値となるエントリです。

戻り値

表 3-41 GET\_DN ファンクションの戻り値

値	説明
VARCHAR2 (ファンクション戻り値)	エントリの PL/SQL 文字列での X.500 識別名です。 問題があった場合の戻り値は NULL です。

例外

表 3-42 GET\_DN ファンクションの例外

例外	説明
invalid_session	セッション・ハンドル ld が無効な場合に呼び出されます。
invalid_message	受信した msg ハンドルが無効な場合に呼び出されます。
get_dn_error	識別名 (DN) を確認中に問題があった場合に呼び出されます。

### **使用方法**

`get_dn()` ファンクションを使用すると、プログラム・ロジックが結果セット全体にわたって同一処理を反復する間に、エントリの識別名 (DN) を取り出せます。また、この識別名 (DN) を `explode_dn()` への入力として使用すると、その DN の個々の構成要素を取り出せます。

### **関連項目**

`DBMS_LDAP.explode_dn()`

## get\_values ファンクション

get\_values() ファンクションを使用すると、特定のエントリの特定の属性に関連する値をすべて取り出せます。

### 構文

```
FUNCTION get_values
(
    ld    IN SESSION,
    ldapentry IN MESSAGE,
    attr  IN VARCHAR2
)
RETURN STRING_COLLECTION;
```

### パラメータ

表 3-43 GET\_VALUES ファンクションのパラメータ

パラメータ	説明
ld	有効な LDAP セッション・ハンドルです。
ldapentry	検索結果から戻されたエントリの有効なハンドルです。
attr	値を検索する対象となる属性の名前です。

### 戻り値

表 3-44 GET\_VALUES ファンクションの戻り値

値	説明
STRING_COLLECTION (ファンクション戻り値)	指定した属性のすべての値を含む PL/SQL 文字列の集合です。 指定した属性に関連する値がない場合の戻り値は NULL です。

### 例外

表 3-45 GET\_VALUES ファンクションの例外

例外	説明
invalid session	セッション・ハンドル ld が無効な場合に呼び出されます。
invalid message	受信したエントリのハンドルが無効な場合に呼び出されます。

**使用方法**

`get_values()` ファンクションは、最初に `first_entry()` または `next_entry()` のコールでエントリのハンドルを取り出してからコールしてください。属性の名前は、事前に判別している場合もあれば、`first_attribute()` または `next_attribute()` のコールによって初めて判別する場合もあります。`get_values()` ファンクションでは、取り出す属性のデータ型は常に文字列であるとみなされます。バイナリ・データ型を取り出すには、`get_values_len()` を使用する必要があります。

**関連項目**

`DBMS_LDAP.first_entry()`、`DBMS_LDAP.next_entry()`、`DBMS_LDAP.count_values()`、`DBMS_LDAP.get_values_len()`

## get\_values\_len ファンクション

get\_values\_len() ファンクションを使用すると、バイナリ構文を持つ属性の値を取り出せます。

### 構文

```
FUNCTION get_values_len
(
    ld      IN SESSION,
    ldapentry IN MESSAGE,
    attr IN VARCHAR2
)
RETURN BINVAL_COLLECTION;
```

### パラメータ

表 3-46 GET\_VALUES\_LEN ファンクションのパラメータ

パラメータ	説明
ld	有効な LDAP セッション・ハンドルです。
ldapentrymsg	検索結果から戻されたエントリの有効なハンドルです。
attr	値の検索対象となる属性の文字列名です。

### 戻り値

表 3-47 GET\_VALUES\_LEN ファンクションの戻り値

値	説明
BINVAL_COLLECTION (ファンクション戻り値)	指定した属性のすべての値を含む PL/SQL RAW 型データ' の集合です。  指定した属性に関連する値がない場合の戻り値は NULL です。

### 例外

表 3-48 GET\_VALUES\_LEN ファンクションの例外

例外	説明
invalid_session	セッション・ハンドル ld が無効な場合に呼び出されます。
invalid_message	受信したエントリのハンドルが無効な場合に呼び出されます。

### 使用方法

`get_values_len()` ファンクションは、`first_entry()` または `next_entry()` のコールでエントリのハンドルを取り出してからコールしてください。属性の名前は、事前に判別している場合もあれば、`first_attribute()` または `next_attribute()` のコールによって初めて判別する場合もあります。このファンクションは、バイナリの属性値および非バイナリの属性値の取出しに使用できます。

### 関連項目

`DBMS_LDAP.first_entry()`、`DBMS_LDAP.next_entry()`、`DBMS_LDAP.count_values_len()`、`DBMS_LDAP.get_values()`

## delete\_s ファンクション

delete\_s() ファンクションを使用すると、LDAP ディレクトリ情報ツリー内のリーフ・エントリを削除できます。

### 構文

```
FUNCTION delete_s
(
    ld          IN SESSION,
    entrydn IN VARCHAR2
)
RETURN PLS_INTEGER;
```

### パラメータ

表 3-49 DELETE\_S ファンクションのパラメータ

パラメータ名	説明
ld	有効な LDAP セッションです。
entrydn	削除するエントリの X.500 識別名です。

### 戻り値

表 3-50 DELETE\_S ファンクションの戻り値

値	説明
PLS_INTEGER (ファンクション戻り値)	削除操作が正常終了した場合の戻り値は DBMS_LDAP.SUCCESS です。その他の場合は例外が呼び出されます。

### 例外

表 3-51 DELETE\_S ファンクションの例外

例外	説明
invalid_session	セッション・ハンドル ld が無効な場合に呼び出されます。
invalid_entry_dn	エントリの識別名が無効な場合に呼び出されます。
general_error	その他すべてのエラーに対応します。この例外に関連するエラー文字列で、エラーの詳細が説明されます。

### 使用方法

`delete_s()` ファンクションを使用すると、LDAP DIT 内のリーフ・レベルのエントリのみを削除できます。リーフ・レベルのエントリとは、下位に子エントリまたは `ldap` エントリを持たないエントリのことです。このファンクションは、リーフ以外のエントリの削除には使用できません。

### 関連項目

`DBMS_LDAP.modrdn2_s()`

## modrdn2\_s ファンクション

modrdn2\_s() ファンクションを使用すると、エントリの相対識別名を変更できます。

### 構文

```
FUNCTION modrdn2_s
(
    ld IN SESSION,
    entrydn IN VARCHAR2
    newrdn IN VARCHAR2
    deleteoldrdn IN PLS_INTEGER
)
RETURN PLS_INTEGER;
```

### パラメータ

表 3-52 MODRDN2\_S ファンクションのパラメータ

パラメータ	説明
ld	有効な LDAP セッション・ハンドルです。
entrydn	エントリの識別名です（このエントリは DIT 内のリーフ・ノードです）。
newrdn	エントリの新しい相対識別名です。
deleteoldrdn	0（ゼロ）以外にした場合は、古い名前から引き継いだ属性値をエントリから削除する必要があることを示すブール値です。

### 戻り値

表 3-53 MODRDN2\_S ファンクションの戻り値

値	説明
PLS_INTEGER（ファンクション戻り値）	操作が正常終了した場合の戻り値は DBMS_LDAP.SUCCESS です。その他の場合は例外が呼び出されます。

例外

表 3-54 MODRDN2\_S ファンクションの例外

例外	説明
invalid_session	セッション・ハンドルldが無効な場合に呼び出されます。
invalid_entry_dn	エントリの識別名が無効な場合に呼び出されます。
invalid_rdn	LDAP 相対識別名（RDN）が無効です。
invalid_deleteoldrdn	LDAP deleteoldrdnが無効です。
general_error	その他すべてのエラーに対応します。この例外に関連するエラー文字列で、エラーの詳細が説明されます。

使用方法

nodrdn2\_s() ファンクションを使用すると、DIT のリーフ・ノードの名前を変更できます。認識の指標となる相対識別名のみが変更されます。LDAP v3 規格でこの関数を使用しないでください。同一の基本機能を持つ rename\_s() を使用してください。

関連項目

DBMS\_LDAP.rename\_s()

## err2string ファンクション

err2string() ファンクションを使用すると、LDAP エラー・コードを、API の動作環境で使用されている各国語の文字列に変換できます。

### 構文

```
FUNCTION err2string
(
    ldap_err IN PLS_INTEGER
)
RETURN VARCHAR2;
```

### パラメータ

表 3-55 ERR2STRING ファンクションのパラメータ

パラメータ	説明
ldap_err	いずれかの API コールから戻されたエラー番号です。

### 戻り値

表 3-56 ERR2STRING ファンクションの戻り値

値	説明
VARCHAR2 (ファンクション戻り値)	各国語へ適切に変換された文字列です。この文字列で、エラーの詳細が説明されます。

### 例外

表 3-57 ERR2STRING ファンクションの例外

例外	説明
N/A	ありません。

### 使用方法

このリリースでは、API コールにエラーが発生した場合、例外処理メカニズムによって自動的にこの関数がコールされます。

### 関連項目

N/A

create\_mod\_array ファンクション

create\_mod\_array() ファンクションを使用すると、modify\_s() ファンクションまたは add\_s() ファンクションを使用してエントリに適用される変更配列に、メモリーが割り当てられます。

構文

```
FUNCTION create_mod_array
(
    num IN PLS_INTEGER
)
RETURN MOD_ARRAY;
```

パラメータ

表 3-58 CREATE\_MOD\_ARRAY ファンクションのパラメータ

パラメータ	説明
num	追加または変更する属性の数。

戻り値

表 3-59 CREATE\_MOD\_ARRAY ファンクションの戻り値

値	説明
MOD_ARRAY (ファンクション戻り値)	このデータ構造により、LDAP 変更配列へのポインタが保持されます。 問題があった場合の戻り値は NULL です。

例外

表 3-60 CREATE\_MOD\_ARRAY ファンクションの例外

例外	説明
N/A	LDAP 固有の例外は呼び出されません。

使用方法

このファンクションは、DBMS\_LDAP.add\_s および DBMS\_LDAP.modify\_s を使用するための準備段階の 1 つです。add\_s または modify\_s のコールが終了した後にメモリーを解放するには、DBMS\_LDAP.free\_mod\_array をコールする必要があります。

**関連項目**

DBMS\_LDAP.populate\_mod\_array()、DBMS\_LDAP.modify\_s()、DBMS\_LDAP.add\_s()、DBMS\_LDAP.free\_mod\_array()

populate\_mod\_array プロシージャ（文字列バージョン）

追加操作または変更操作に、1 組の属性情報を代入します。

構文

```
PROCEDURE populate_mod_array
(
    modptr    IN DBMS_LDAP.MOD_ARRAY,
    mod_op    IN PLS_INTEGER,
    mod_type  IN VARCHAR2,
    modval    IN DBMS_LDAP.STRING_COLLECTION
);
```

パラメータ

表 3-61 POPULATE\_MOD\_ARRAY（文字列バージョン） プロシージャのパラメータ

パラメータ	説明
modptr	このデータ構造により、LDAP 変更配列へのポインタが保持されます。
mod_op	このフィールドで、実行する変更の型を指定します。
mod_type	このフィールドで、変更を適用する属性型の名前を指定します。
modval	このフィールドで、追加、削除または置換する属性値を指定します。対象は文字列値のみです。

戻り値

表 3-62 POPULATE\_MOD\_ARRAY（文字列バージョン） プロシージャの戻り値

値	説明
N/A	

## 例外

**表 3-63 POPULATE\_MOD\_ARRAY (文字列バージョン) プロシージャの例外**

例外	説明
invalid_mod_array	LDAP 変更配列が無効です。
invalid_mod_option	LDAP 変更オプションが無効です。
invalid_mod_type	LDAP 変更型が無効です。
invalid_mod_value	LDAP 変更値が無効です。

## 使用方法

このプロシージャは、DBMS\_LDAP.add\_s および DBMS\_LDAP.modify\_s を使用するための準備段階の 1 つです。DBMS\_LDAP.create\_mod\_array をコールした後に、このプロシージャをコールする必要があります。

## 関連項目

DBMS\_LDAP.create\_mod\_array()、DBMS\_LDAP.modify\_s()、DBMS\_LDAP.add\_s()、DBMS\_LDAP.free\_mod\_array()

populate\_mod\_array プロシージャ (バイナリ・バージョン)

追加操作または変更操作に、1 組の属性情報を代入します。DBMS\_LDAP.create\_mod\_array() をコールした後に、このプロシージャをコールする必要があります。

構文

```
PROCEDURE populate_mod_array
(
    modptr    IN DBMS_LDAP.MOD_ARRAY,
    mod_op    IN PLS_INTEGER,
    mod_type  IN VARCHAR2,
    modval    IN DBMS_LDAP.BERVAL_COLLECTION
);
```

パラメータ

表 3-64 POPULATE\_MOD\_ARRAY (バイナリ・バージョン) プロシージャのパラメータ

パラメータ	説明
modptr	このデータ構造により、LDAP 変更配列へのポインタが保持されます。
mod_op	このフィールドで、実行する変更の型を指定します。
mod_type	このフィールドで、変更を適用する属性型の名前を指定します。
modval	このフィールドで、追加、削除または置換する属性値を指定します。対象はバイナリ値のみです。

戻り値

表 3-65 POPULATE\_MOD\_ARRAY (バイナリ・バージョン) プロシージャの戻り値

値	説明
N/A	

## 例外

**表 3-66 POPULATE\_MOD\_ARRAY (バイナリ・バージョン) プロシージャの例外**

例外	説明
invalid_mod_array	LDAP 変更配列が無効です。
invalid_mod_option	LDAP 変更オプションが無効です。
invalid_mod_type	LDAP 変更型が無効です。
invalid_mod_value	LDAP 変更値が無効です。

## 使用方法

このプロシージャは、DBMS\_LDAP.add\_s および DBMS\_LDAP.modify\_s を使用するための準備段階の 1 つです。DBMS\_LDAP.create\_mod\_array をコールした後に、このプロシージャをコールする必要があります。

## 関連項目

DBMS\_LDAP.create\_mod\_array()、DBMS\_LDAP.modify\_s()、DBMS\_LDAP.add\_s()、DBMS\_LDAP.free\_mod\_array()

modify\_s ファンクション

既存の LDAP ディレクトリ・エントリの同期変更を実行します。

構文

```
FUNCTION modify_s
(
    ld          IN DBMS_LDAP.SESSION,
    entrydn     IN VARCHAR2,
    modptr      IN DBMS_LDAP.MOD_ARRAY
)
RETURN PLS_INTEGER;
```

パラメータ

表 3-67 MODIFY\_S ファンクションのパラメータ

パラメータ	説明
ld	このパラメータは LDAP セッションのハンドルで、DBMS_LDAP.init() のコールが正常終了した場合の戻り値と同一です。
entrydn	このパラメータで、内容を変更するディレクトリ・エントリの名前を指定します。
modptr	このパラメータは LDAP 変更構造体のハンドルで、DBMS_LDAP.create_mod_array() のコールが正常終了した場合の戻り値と同一です。

戻り値

表 3-68 MODIFY\_S ファンクションの戻り値

値	説明
PLS_INTEGER	変更操作の成功または失敗を示します。

例外

表 3-69 MODIFY\_S ファンクションの例外

例外	説明
invalid_session	LDAP セッションが無効です。
invalid_entry_dn	LDAP エントリ識別名 (DN) が無効です。
invalid_mod_array	LDAP 変更配列が無効です。

**使用方法**

このファンクションは、`DBMS_LDAP.create_mod_array()` および `DBMS_LDAP.populate_mod_array()` のコールが正常終了した後にコールする必要があります。

**関連項目**

`DBMS_LDAP.create_mod_array()`、`DBMS_LDAP.populate_mod_array()`、`DBMS_LDAP.add_s()`、`DBMS_LDAP.free_mod_array()`

add\_s ファンクション

LDAP ディレクトリに新規エントリを同期的に追加します。add\_s をコールする前に、まず DBMS\_LDAP.create\_mod\_array() と DBMS\_LDAP.populate\_mod\_array() をコールする必要があります。

構文

```
FUNCTION add_s
(
    ld          IN DBMS_LDAP.SESSION,
    entrydn     IN VARCHAR2,
    modptr      IN DBMS_LDAP.MOD_ARRAY
)
RETURN PLS_INTEGER;
```

パラメータ

表 3-70 ADD\_S ファンクションのパラメータ

パラメータ	説明
ld	このパラメータは LDAP セッションのハンドルで、DBMS_LDAP.init() のコールが正常終了した場合の戻り値と同一です。
entrydn	このパラメータで、作成するディレクトリ・エントリの名前を指定します。
modptr	このパラメータは LDAP 変更構造体のハンドルで、DBMS_LDAP.create_mod_array() のコールが正常終了した場合の戻り値と同一です。

戻り値

表 3-71 ADD\_S ファンクションの戻り値

値	説明
PLS_INTEGER	変更操作の成功または失敗を示します。

## 例外

**表 3-72 ADD\_S ファンクションの例外**

例外	説明
invalid_session	LDAP セッションが無効です。
invalid_entry_dn	LDAP エントリ識別名 (DN) が無効です。
invalid_mod_array	LDAP 変更配列が無効です。

## 使用方法

追加するエントリの親エントリは、ディレクトリ内にすでに存在する必要があります。  
このファンクションは、`DBMS_LDAP.create_mod_array()` および `DBMS_LDAP.populate_mod_array()` のコールが正常終了した後にコールする必要があります。

## 関連項目

`DBMS_LDAP.create_mod_array()`、`DBMS_LDAP.populate_mod_array()`、`DBMS_LDAP.modify_s()`、`DBMS_LDAP.free_mod_array()`

free\_mod\_array プロシージャ

DBMS\_LDAP.create\_mod\_array() によって割り当てられたメモリーを解放します。

構文

```
PROCEDURE free_mod_array
(
    modptr IN DBMS_LDAP.MOD_ARRAY
);
```

パラメータ

表 3-73 FREE\_MOD\_ARRAY プロシージャのパラメータ

パラメータ	説明
modptr	このパラメータは LDAP 変更構造体のハンドルで、DBMS_LDAP.create_mod_array() のコールが正常終了した場合の戻り値と同一です。

戻り値

表 3-74 FREE\_MOD\_ARRAY プロシージャの戻り値

値	説明
N/A	

例外

表 3-75 FREE\_MOD\_ARRAY プロシージャの例外

例外	説明
N/A	LDAP 固有の例外は呼び出されません。

使用方法

N/A

関連項目

DBMS\_LDAP.populate\_mod\_array(), DBMS\_LDAP.modify\_s(), DBMS\_LDAP.add\_s(), DBMS\_LDAP.create\_mod\_array()

## count\_values ファンクション

DBMS\_LDAP.get\_values() によって戻された値の数をカウントします。

### 構文

```
FUNCTION count_values  
(  
    values IN DBMS_LDAP.STRING_COLLECTION  
)  
    RETURN PLS_INTEGER;
```

### パラメータ

表 3-76 COUNT\_VALUES ファンクションのパラメータ

パラメータ	説明
values	文字列値の集合です。

### 戻り値

表 3-77 COUNT\_VALUES ファンクションの戻り値

値	説明
PLS_INTEGER	操作の成功または失敗を示します。

### 例外

表 3-78 COUNT\_VALUES ファンクションの例外

例外	説明
N/A	LDAP 固有の例外は呼び出されません。

### 使用方法

N/A

### 関連項目

DBMS\_LDAP.count\_values\_len()、DBMS\_LDAP.get\_values()

count\_values\_len ファンクション

DBMS\_LDAP.get\_values\_len() によって戻された値の数をカウントします。

構文

```
FUNCTION count_values_len
(
    values IN DBMS_LDAP.BINVAL_COLLECTION
)
RETURN PLS_INTEGER;
```

パラメータ

表 3-79 COUNT\_VALUES\_LEN ファンクションのパラメータ

パラメータ	説明
values	バイナリ値の集合です。

戻り値

表 3-80 COUNT\_VALUES\_LEN ファンクションの戻り値

値	説明
PLS_INTEGER	操作の成功または失敗を示します。

例外

表 3-81 COUNT\_VALUES\_LEN ファンクションの例外

例外	説明
N/A	LDAP 固有の例外は呼び出されません。

使用方法

N/A

関連項目

DBMS\_LDAP.count\_values(), DBMS\_LDAP.get\_values\_len()

## rename\_s ファンクション

LDAP エントリの名前を同期的に変更します。

### 構文

```
FUNCTION rename_s
(
    ld          IN SESSION,
    dn          IN VARCHAR2,
    newrdn      IN VARCHAR2,
    ewparent    IN VARCHAR2,
    deleteoldrdn IN PLS_INTEGER,
    serverctrls IN LDAPCONTROL,
    clientctrls IN LDAPCONTROL
)
    RETURN PLS_INTEGER;
```

### パラメータ

表 3-82 RENAME\_S ファンクションのパラメータ

パラメータ	説明
ld	このパラメータは LDAP セッションのハンドルで、DBMS_LDAP.init() のコールが正常終了した場合の戻り値と同一です。
dn	このパラメータで、改名または移動するディレクトリ・エントリの名前を指定します。
newrdn	このパラメータで、新しい相対識別名 (RDN) を指定します。
newparent	このパラメータで、新しい親の識別名 (DN) を指定します。
deleteoldrdn	このパラメータで、古い相対識別名 (RDN) を保持するかどうかを指定します。この値を 1 にすると、古い RDN が削除されます。
serverctrls	現在はサポートされていません。
clientctrls	現在はサポートされていません。

### 戻り値

表 3-83 RENAME\_S ファンクションの戻り値

値	説明
PLS_INTEGER	操作の成功または失敗を示します。

例外

表 3-84 RENAME\_S ファンクションの例外

例外	説明
invalid_session	LDAP セッションが無効です。
invalid_entry_dn	LDAP 識別名 (DN) が無効です。
invalid_rdn	LDAP 相対識別名 (RDN) が無効です。
invalid_newparent	LDAP の新規の親が無効です。
invalid_deleteoldrdn	LDAP deleteoldrdn が無効です。

使用方法

N/A

関連項目

DBMS\_LDAP.modrdn2\_s()

## explode\_dn ファンクション

識別名（DN）を個々の構成要素に分割します。

### 構文

```
FUNCTION explode_dn
(
    dn          IN VARCHAR2,
    notypes     IN PLS_INTEGER
)
RETURN STRING_COLLECTION;
```

### パラメータ

表 3-85 EXPLODE\_DN ファンクションのパラメータ

パラメータ	説明
dn	このパラメータで、分割するディレクトリ・エントリの名前を指定します。
notypes	このパラメータで、属性タグを戻すかどうかを指定します。この値を 0（ゼロ）にすると、属性タグは戻されません。

### 戻り値

表 3-86 EXPLODE\_DN ファンクションの戻り値

値	説明
STRING_COLLECTION	文字列の配列です。識別名（DN）が分割できない場合は NULL が戻されます。

### 例外

表 3-87 EXPLODE\_DN ファンクションの例外

例外	説明
invalid_entry_dn	LDAP 識別名（DN）が無効です。
invalid_notypes	LDAP notypes の値が無効です。

### 使用方法

N/A

**関連項目**

`DBMS_LDAP.get_dn()`

## open\_ssl ファンクション

すでに確立されている LDAP 接続を介して SSL (Secure Sockets Layer) 接続を確立します。

### 構文

```
FUNCTION open_ssl
(
    ld                IN SESSION,
    sslwrl            IN VARCHAR2,
    sslwalletpasswd   IN VARCHAR2,
    sslauth           IN PLS_INTEGER
)
RETURN PLS_INTEGER;
```

### パラメータ

表 3-88 OPEN\_SSL ファンクションのパラメータ

パラメータ	説明
ld	このパラメータは LDAP セッションのハンドルで、DBMS_LDAP.init() のコールが正常終了した場合の戻り値と同一です。
sslwrl	このパラメータで、Wallet の位置を指定します (サーバー、またはクライアントとサーバーの SSL 接続の場合は必須)。
sslwalletpasswd	このパラメータで、Wallet のパスワードを指定します (サーバー、またはクライアントとサーバーの SSL 接続の場合は必須)。
sslauth	このパラメータで、SSL 認証モード (SSL 認証なしの場合は 1、サーバー認証の場合は 2、クライアントとサーバーの認証の場合は 3) を指定します。

### 戻り値

表 3-89 OPEN\_SSL ファンクションの戻り値

値	説明
PLS_INTEGER	操作の成功または失敗を示します。

例外

表 3-90 OPEN\_SSL ファンクションの例外

例外	説明
invalid_session	LDAP セッションが無効です。
invalid_ssl_wallet_loc	LDAP SSL の Wallet の場所が無効です。
invalid_ssl_wallet_passwd	LDAP SSL の Wallet のパスワードが無効です。
invalid_ssl_auth_mode	LDAP SSL 認証モードが無効です。

使用方法

有効な LDAP セッションを取得するには、まず DBMS\_LDAP.init() をコールする必要があります。

関連項目

DBMS\_LDAP.init()

---

## コマンドライン・ツールの構文

この章では、LDAP データ交換フォーマット (LDIF) と LDAP コマンドライン・ツールを使用するための構文、使用方法および使用例を紹介します。この章では、次の項目について説明します。

- [LDAP データ交換フォーマット \(LDIF\) 構文](#)
- [コマンドライン・ツールの構文](#)
- [カタログ管理ツールの構文](#)

# LDAP データ交換フォーマット（LDIF）構文

ディレクトリ・エントリの標準ファイル形式は、次のとおりです。

```
dn: distinguished_name
attribute_type: attribute_value
.
.
.
objectClass: object_class_value
.
.
.
```

プロパティ	値	説明
dn:	RDN,RDN,RDN, ...	相対識別名（RDN）をカンマで区切ります。
attribute:	attribute_value	この行は、エントリの各属性および複数値属性の各属性値ごとに繰り返します。
objectClass:	object_class_value	この行は、各オブジェクト・クラスごとに繰り返します。

次の例は、従業員のファイル・エントリを示しています。1 行目は識別名（DN）です。DN に続く各行は、属性のニーモニックで始まり、その属性の値が続きます。各エントリが、そのエントリのオブジェクト・クラスを定義する行で終了していることに注意してください。

```
dn: cn=Suzie Smith,ou=Server Technology,o=Acme, c=US
cn: Suzie Smith
cn: SuzieS
sn: Smith
email: ssmith@us.Acme.com
telephoneNumber: 69332
photo:/ORACLE_HOME/empdir/photog/ssmith.jpg
objectClass: organizational person
objectClass: person
objectClass: top
```

次の例は、組織のファイル・エントリを示しています。

```
dn: o=Acme,c=US
o: Acme
ou: Financial Applications
objectClass: organization
objectClass: top
```

## LDIF 形式化の注意事項

次に形式化規則のリストを示します。このリストは、全規則を網羅しているわけではありません。

- 追加対象のエントリに属しているすべての必須属性は、非 NULL 値で LDIF ファイルに記述する必要があります。

**ヒント：** オブジェクト・クラスの必須属性とオプション属性のタイプを調べるには、Oracle Directory Manager を使用します。『Oracle Internet Directory 管理者ガイド』を参照してください。

- 非表示文字やタブは、Base64 エンコーディングによる属性値で記述します。
- ファイル内のエントリの間は、空白行で区切る必要があります。
- ファイルには、少なくとも 1 つのエントリが含まれている必要があります。
- 次の行に継続する場合は、継続行を空白またはタブで開始します。
- 個々のエントリの間空白行を追加してください。
- 写真などのバイナリ・ファイルは、スラッシュ (/) で始まるファイルの絶対アドレスで参照を付けます。
- 識別名 (DN) には、オブジェクトに対する一意の完全なディレクトリ・アドレスが含まれます。
- DN の後にリストされる行には、属性とその値が含まれます。入力ファイルで使われる DN と属性は、DIT の既存の構造と一致している必要があります。DIT 内で実装していない属性は、入力ファイルで使わないでください。
- LDIF ファイル内のエントリは、DIT が上位から下位へ作成されるように順に記述します。エントリがその DN の上位のエントリに依存している場合は、その子エントリの前に上位エントリを必ず追加してください。
- LDIF ファイル内にスキーマを定義するときは、左カッコと最初のテキストの間、および最後のテキストと右カッコの間に空白を挿入してください。

**関連項目：** LDIF 形式化規則および LDIF ファイルでの NLS の使用方法については、『Oracle Internet Directory 管理者ガイド』に記載されている各種資料を参照してください。

## コマンドライン・ツールの構文

この項では、次のツールの使用方法を説明します。

- [ldapadd 構文](#)
- [ldapaddmt 構文](#)
- [ldapbind 構文](#)
- [ldapcompare 構文](#)
- [ldapdelete 構文](#)
- [ldapmoddn 構文](#)
- [ldapmodify 構文](#)
- [ldapmodifymt 構文](#)
- [ldapsearch 構文](#)

### ldapadd 構文

ldapadd コマンドライン・ツールを使用して、エントリとそのオブジェクト・クラス、属性および値をディレクトリに追加できます。既存のエントリに属性を追加するには、ldapmodify コマンドを使用します。ldapmodify コマンドは、4-12 ページの「[ldapmodify 構文](#)」を参照してください。

**関連項目：** 入力ファイルを使用してサーバーを構成するために ldapadd を使用する方法は、『Oracle Internet Directory 管理者ガイド』を参照してください。

ldapadd は、次の構文で使います。

```
ldapadd [arguments] -f filename
```

*filename* は、4-2 ページの「[LDAP データ交換フォーマット \(LDIF\) 構文](#)」で説明する仕様に従って作成された、LDIF ファイルの名前です。

次の例では、LDIF ファイル `my_ldif_file.ldi` 内に指定されたエントリをディレクトリに追加しています。

```
ldapadd -p 389 -h myhost -f my_ldif_file.ldi
```

オプションの引数	説明
-b	ファイルにバイナリ・ファイル名が含まれていることを指定します。バイナリ・ファイル名はスラッシュで始まります。ツールは、参照先のファイルから実際の値を取り出します。
-c	エラーが発生しても処理を継続する場合に指定します。エラーはレポートされます。(このオプションを使用しない場合、エラーが発生すると <code>ldapadd</code> は停止します。)
-D <i>binddn</i>	ディレクトリに対して認証するときに、 <i>binddn</i> に指定されているエントリとして認証することを指定します。この引数は、 <i>-w password</i> オプションとともに使用します。
-E " <i>character_set</i> "	ネイティブ・キャラクタ・セット・エンコーディングを指定します。『Oracle Internet Directory 管理者ガイド』の NLS の章を参照してください。
-f <i>filename</i>	LDIF 形式のインポート・データ・ファイルの入力名を指定します。LDIF ファイルのフォーマット方法の詳細は、4-2 ページの「 <a href="#">LDAP データ交換フォーマット (LDIF) 構文</a> 」を参照してください。
-h <i>ldaphost</i>	デフォルトのホスト（ローカル・コンピュータ）ではなく、 <i>ldaphost</i> に接続します。 <i>ldaphost</i> には、コンピュータ名または IP アドレスを指定します。
-K	-k と同様ですが、Kerberos バインドの最初のステップのみ実行します。
-k	簡易認証のかわりに、Kerberos 認証を使用して認証します。このオプションを使用可能にするには、定義済みの Kerberos でコンパイルする必要があります。  有効なチケット認可チケットをすでに所有している必要があります。
-n	操作を実際には実行せずに、予測結果を示します。
-p <i>ldapport</i>	TCP ポート <i>ldapport</i> 上のディレクトリに接続します。このオプションを指定しない場合は、デフォルト・ポート（389）に接続されます。
-P <i>wallet_password</i>	Wallet のパスワードを指定します（サーバー、またはクライアントとサーバーの SSL 接続の場合は必須）。
-U <i>SSLAuth</i>	SSL 認証モードを指定します。  <ul style="list-style-type: none"> <li>■ 1: SSL 認証なし</li> <li>■ 2: サーバー認証</li> <li>■ 3: クライアントとサーバーの認証</li> </ul>
-v	冗長モードを指定します。

オプションの引数	説明
-w <i>password</i>	接続に必要なパスワードを指定します。
-W <i>wallet_location</i>	Wallet の位置を指定します（サーバー、またはクライアントとサーバーの SSL 接続の場合は必須）。

ldapaddmt 構文

ldapaddmt は ldapadd と類似しています。ldapaddmt を使用して、エントリとそのオブジェクト・クラス、属性および値をディレクトリに追加できます。ldapadd と異なるのは、複数のエントリを同時に追加するために複数のスレッドをサポートしている点です。

LDIF エントリの処理中に、ldapaddmt は、カレント・ディレクトリ内の add.log ファイルにエラー・ログを記録します。

ldapaddmt は、次の構文で使います。

```
ldapaddmt -T number_of_threads -h host -p port -f filename
```

*filename* は、4-2 ページの「LDAP データ交換フォーマット (LDIF) 構文」で説明する仕様に従って作成された、LDIF ファイルの名前です。

次の例は、5 つの同時スレッドを使用して、ファイル myentries.ldif 内のエントリを処理しています。

```
ldapaddmt -T 5 -h node1 -p 3000 -f myentries.ldif
```

**注意：** 同時スレッドの数が増加すると、LDIF エントリの作成は速くなりますが、システム・リソースはより多く消費されます。

オプションの引数	説明
-b	データ・ファイルにバイナリ・ファイル名が含まれていることを指定します。バイナリ・ファイル名はスラッシュで始まります。ツールは、参照先のファイルから実際の値を取り出します。
-c	エラーが発生しても処理を継続する場合に指定します。エラーはレポートされます。（このオプションを使用しない場合、エラーが発生するとツールは停止します。）
-D <i>binddn</i>	ディレクトリに対して認証するときに、 <i>binddn</i> に指定されているエントリとして認証することを指定します。この引数は、-w <i>password</i> オプションとともに使います。

オプションの引数	説明
-E <i>"character_set"</i>	ネイティブ・キャラクタ・セット・エンコーディングを指定します。『Oracle Internet Directory 管理者ガイド』の NLS の章を参照してください。
-h <i>ldaphost</i>	デフォルトのホスト（ローカル・コンピュータ）ではなく、 <i>ldaphost</i> に接続します。 <i>ldaphost</i> には、コンピュータ名または IP アドレスを指定します。
-K	-k と同様ですが、Kerberos バインドの最初のステップのみ実行します。
-k	簡易認証のかわりに、Kerberos 認証を使用して認証します。このオプションを使用可能にするには、定義済みの Kerberos でコンパイルする必要があります。 有効なチケット認可チケットをすでに所有している必要があります。
-n	操作を実際には実行せずに、予測結果を示します。
-p <i>ldapport</i>	TCP ポート <i>ldapport</i> 上のディレクトリに接続します。このオプションを指定しない場合は、デフォルト・ポート（389）に接続されます。
-P <i>wallet_password</i>	Wallet のパスワードを指定します（サーバー、またはクライアントとサーバーの SSL 接続の場合は必須）。
-T	エントリを同時に処理するスレッドの数を設定します。
-U <i>SSLAuth</i>	SSL 認証モードを指定します。 <ul style="list-style-type: none"><li>■ 1: SSL 認証なし</li><li>■ 2: サーバー認証</li><li>■ 3: クライアントとサーバーの認証</li></ul>
-v	冗長モードを指定します。
-w <i>password</i>	接続に必要なパスワードを指定します。
-W <i>wallet_location</i>	Wallet の位置を指定します（サーバー、またはクライアントとサーバーの SSL 接続の場合は必須）。

## ldapbind 構文

ldapbind コマンドライン・ツールを使用して、サーバーに対してクライアントを認証できるかどうかを調べることができます。

ldapbind は、次の構文で使います。

```
ldapbind [arguments]
```

オプションの引数	説明
-D <i>binddn</i>	ディレクトリに対して認証するときに、 <i>binddn</i> に指定されているエントリとして認証することを指定します。この引数は、 <i>-w password</i> オプションとともに使用します。
-E " <i>.character_set</i> "	ネイティブ・キャラクタ・セット・エンコーディングを指定します。『Oracle Internet Directory 管理者ガイド』の NLS の章を参照してください。
-h <i>ldaphost</i>	デフォルトのホスト（ローカル・コンピュータ）ではなく、 <i>ldaphost</i> に接続します。 <i>ldaphost</i> には、コンピュータ名または IP アドレスを指定します。
-n	操作を実際には実行せずに、予測結果を示します。
-p <i>ldapport</i>	TCP ポート <i>ldapport</i> 上のディレクトリに接続します。このオプションを指定しない場合は、デフォルト・ポート（389）に接続されます。
-P <i>wallet_password</i>	Wallet のパスワードを指定します（サーバー、またはクライアントとサーバーの SSL 接続の場合は必須）。
-U <i>SSLAuth</i>	SSL 認証モードを指定します。 <ul style="list-style-type: none"><li>■ 1: SSL 認証なし</li><li>■ 2: サーバー認証</li><li>■ 3: クライアントとサーバーの認証</li></ul>
-w <i>password</i>	接続に必要なパスワードを指定します。
-W <i>wallet_location</i>	Wallet の位置を指定します（サーバー、またはクライアントとサーバーの SSL 接続の場合は必須）。

ldapcompare 構文

ldapcompare コマンドライン・ツールを使用して、コマンドラインで指定した属性値と、ディレクトリ・エントリの属性値を比較できます。

ldapcompare は、次の構文で使用します。

ldapcompare [*arguments*]

次の例は、Person Nine のタイトルが associate であるかどうかを通知します。

```
ldapcompare -p 389 -h myhost -b "cn=Person Nine, ou=EuroSInet Suite, o=IMC, c=US" -a title -v associate
```

必須の引数	説明
<code>-a attribute name</code>	比較を実行する属性を指定します。
<code>-b basedn</code>	比較を実行するエントリの識別名を指定します。
<code>-v attribute value</code>	比較する属性値を指定します。

オプションの引数	説明
<code>-D binddn</code>	ディレクトリに対して認証するとき、 <code>binddn</code> に指定されているエントリとして認証することを指定します。この引数は、 <code>-w password</code> オプションとともに使用します。
<code>-d debug-level</code>	デバッグ・レベルを設定します。『Oracle Internet Directory 管理者ガイド』のディレクトリ・サーバーの管理の章を参照してください。
<code>-E "character_set"</code>	ネイティブ・キャラクタ・セット・エンコーディングを指定します。『Oracle Internet Directory 管理者ガイド』の NLS の章を参照してください。
<code>-f filename</code>	入力ファイル名を指定します。
<code>-h ldaphost</code>	デフォルトのホスト（ローカル・コンピュータ）ではなく、 <code>ldaphost</code> に接続します。 <code>ldaphost</code> には、コンピュータ名または IP アドレスを指定します。
<code>-p ldapport</code>	TCP ポート <code>ldapport</code> 上のディレクトリに接続します。このオプションを指定しない場合は、デフォルト・ポート（389）に接続されます。
<code>-P wallet_password</code>	Wallet のパスワードを指定します（サーバー、またはクライアントとサーバーの SSL 接続の場合は必須）。
<code>-U SSLAuth</code>	SSL 認証モードを指定します。 <ul style="list-style-type: none"> <li>■ 1: SSL 認証なし</li> <li>■ 2: サーバー認証</li> <li>■ 3: クライアントとサーバーの認証</li> </ul>
<code>-w password</code>	接続に必要なパスワードを指定します。
<code>-W wallet_location</code>	Wallet の位置を指定します（サーバー、またはクライアントとサーバーの SSL 接続の場合は必須）。

ldapdelete 構文

ldapdelete コマンドライン・ツールを使用して、コマンドラインに指定したディレクトリからエントリ全体を削除できます。

ldapdelete は、次の構文で使います。

```
ldapdelete [arguments] "entry_DN"
```

次の例では、myhost という名前のホストでポート 389 を使用しています。

```
ldapdelete -p 389 -h myhost "ou=EuroSInet Suite, o=IMC, c=US"
```

オプションの引数	説明
-D binddn	ディレクトリに対して認証するときに、binddn パラメータに完全識別名 (DN) を使用します。通常、-w password オプションとともに使用されます。
-d debug-level	デバッグ・レベルを設定します。『Oracle Internet Directory 管理者ガイド』のディレクトリ・サーバーの管理の章を参照してください。
-E "character_set"	ネイティブ・キャラクタ・セット・エンコーディングを指定します。『Oracle Internet Directory 管理者ガイド』の NLS の章を参照してください。
-f filename	入力ファイル名を指定します。
-h ldaphost	デフォルトのホスト（ローカル・コンピュータ）ではなく、ldaphost に接続します。ldaphost には、コンピュータ名または IP アドレスを指定します。
-k	簡易認証のかわりに、Kerberos 認証を使用して認証します。このオプションを使用可能にするには、定義済みの Kerberos でコンパイルする必要があります。  有効なチケット認可チケットをすでに所有している必要があります。
-n	削除を実際には実行せずに、予測結果を示します。
-p ldapport	TCP ポート ldapport 上のディレクトリに接続します。このオプションを指定しない場合は、デフォルト・ポート（389）に接続されます。
-P wallet_password	Wallet のパスワードを指定します（サーバー、またはクライアントとサーバーの SSL 接続の場合は必須）。
-U SSLAuth	SSL 認証モードを指定します。 <ul style="list-style-type: none"><li>1: SSL 認証なし</li><li>2: サーバー認証</li><li>3: クライアントとサーバーの認証</li></ul>

オプションの引数	説明
<code>-v</code>	冗長モードを指定します。
<code>-w <i>password</i></code>	接続に必要なパスワードを指定します。
<code>-W <i>wallet_location</i></code>	Wallet の位置を指定します（サーバー、またはクライアントとサーバーの SSL 接続の場合は必須）。

## ldapmoddn 構文

ldapmoddn コマンドライン・ツールを使用して、エントリの識別名（DN）または相対識別名（RDN）を変更できます。

ldapmoddn は、次の構文で使います。

```
ldapmoddn [arguments]
```

次の例では、ldapmoddn を使用して、DN の RDN コンポーネントを "cn=dcpl" から "cn=thanh mai" に変更しています。ポートは 389、myhost という名前のホストを使用しています。

```
ldapmoddn -p 389 -h myhost -b "cn=dcpl,dc=Americas,dc=imc,dc=com" -R "cn=thanh mai"
```

必須の引数	説明
<code>-b <i>basedn</i></code>	変更されるエントリの識別名（DN）を指定します。

オプションの引数	説明
<code>-D <i>binddn</i></code>	ディレクトリに対して認証するときに、 <i>binddn</i> に指定されているエントリとして認証します。この引数は、 <code>-w <i>password</i></code> オプションとともに使います。
<code>-E "<i>character_set</i>"</code>	ネイティブ・キャラクタ・セット・エンコーディングを指定します。『Oracle Internet Directory 管理者ガイド』の NLS の章を参照してください。
<code>-f <i>filename</i></code>	入力ファイル名を指定します。
<code>-h <i>ldaphost</i></code>	デフォルトのホスト（ローカル・コンピュータ）ではなく、 <i>ldaphost</i> に接続します。 <i>ldaphost</i> には、コンピュータ名または IP アドレスを指定します。
<code>-N <i>newparent</i></code>	RDN の新しい親を指定します。

オプションの引数	説明
-p <i>ldapport</i>	TCP ポート <i>ldapport</i> 上のディレクトリに接続します。このオプションを指定しない場合は、デフォルト・ポート（389）に接続されます。
-P <i>wallet_password</i>	Wallet のパスワードを指定します（サーバー、またはクライアントとサーバーの SSL 接続の場合は必須）。
-r	旧 RDN を変更エントリ内に値として保持しないことを指定します。この引数が指定されない場合、旧 RDN は変更エントリ内に属性として保持されます。
-R <i>newrdn</i>	新規 RDN を指定します。
-U <i>SSLAuth</i>	SSL 認証モードを指定します。 <ul style="list-style-type: none"><li>1: SSL 認証なし</li><li>2: サーバー認証</li><li>3: クライアントとサーバーの認証</li></ul>
-w <i>password</i>	接続に必要なパスワードを指定します。
-W <i>wallet_location</i>	Wallet の位置を指定します（サーバー、またはクライアントとサーバーの SSL 接続の場合は必須）。

ldapmodify 構文

ldapmodify コマンドライン・ツールを使用して、属性を変更できます。

ldapmodify は、次の構文で使います。

```
ldapmodify [arguments] -f filename
```

*filename* は、4-2 ページの「LDAP データ交換フォーマット (LDIF) 構文」で説明する仕様に従って作成された、LDIF ファイルの名前です。

次の表の引数リストは、すべての引数ではありません。

オプションの引数	説明
-a	エントリが追加対象で、入力ファイルが LDIF 形式であることを示します。
-b	データ・ファイルにバイナリ・ファイル名が含まれていることを指定します。バイナリ・ファイル名はスラッシュで始まります。
-c	エラーが発生しても処理を継続する場合に指定します。エラーはレポートされます。（このオプションを使用しない場合、エラーが発生すると ldapmodify は停止します。）

オプションの引数	説明
<code>-D binddn</code>	ディレクトリに対して認証するときに、 <i>binddn</i> に指定されているエントリとして認証することを指定します。この引数は、 <code>-w password</code> オプションとともに使用します。
<code>-E "character_set"</code>	ネイティブ・キャラクタ・セット・エンコーディングを指定します。『Oracle Internet Directory 管理者ガイド』の NLS の章を参照してください。
<code>-h ldaphost</code>	デフォルトのホスト（ローカル・コンピュータ）ではなく、 <i>ldaphost</i> に接続します。 <i>ldaphost</i> には、コンピュータ名または IP アドレスを指定します。
<code>-n</code>	操作を実際には実行せずに、予測結果を示します。
<code>-p ldapport</code>	TCP ポート <i>ldapport</i> 上のディレクトリに接続します。このオプションを指定しない場合は、デフォルト・ポート（389）に接続されます。
<code>-P wallet_password</code>	Wallet のパスワードを指定します（サーバー、またはクライアントとサーバーの SSL 接続の場合は必須）。
<code>-U SSLAuth</code>	SSL 認証モードを指定します。 <ul style="list-style-type: none"> <li>■ 1: SSL 認証なし</li> <li>■ 2: サーバー認証</li> <li>■ 3: クライアントとサーバーの認証</li> </ul>
<code>-v</code>	冗長モードを指定します。
<code>-w password</code>	デフォルトの非認証の NULL バインドをオーバーライドします。認証を強制するには、このオプションを <code>-D</code> オプションとともに使用します。
<code>-W wallet_location</code>	Wallet の位置を指定します（サーバー、またはクライアントとサーバーの SSL 接続の場合は必須）。

`-f` フラグを使用して `modify`、`delete` および `modifyrdn` 操作を実行するには、入力ファイル形式に LDIF を使用します（4-2 ページの「[LDAP データ交換フォーマット \(LDIF\) 構文](#)」を参照してください）。仕様は次のとおりです。

エントリは常に空白行で区切ります。

属性値の後の空白など、LDIF 入力ファイルにおける不要な空白は、LDAP 操作が失敗する原因となります。

**第1行:** 変更レコードの場合は、その1行目にリテラル `dn:`、その後にエントリの識別名 (DN) 値を記述します。たとえば、次のように記述します。

```
dn:cn=Barbara Fritchey,ou=Sales,o=Oracle,c=US
```

**第2行:** 変更レコードの場合は、その2行目にリテラル `changetype:`、その後に変更の種類 (`add`、`delete`、`modify`、`modrdn` など) を記述します。たとえば、次のように記述します。

```
changetype:modify
```

または

```
changetype:modrdn
```

変更の種類に応じて、次の要件に従って各レコードの残りの部分をフォーマットします。

- `changetype:add`

LDIF 形式を使用します (4-2 ページの「[LDAP データ交換フォーマット \(LDIF\) 構文](#)」を参照してください)。

- `changetype:modify`

この `changetype` に続く行には、前述の第1行で指定したエントリに属する属性に対する変更内容を記述します。属性の変更は、3 種類 (`add`、`delete` および `replace`) を指定できます。それぞれの変更について次に説明します。

- **属性値の追加。** `changetype modify` のこのオプションは、既存の複数値の属性にさらに値を追加します。属性が存在しない場合は、指定した値で新規属性を追加します。

```
add: attribute name
attribute name: value1
attribute name: value2...
```

例:

```
dn:cn=Barbara Fritchey,ou=Sales,o=Oracle,c=US
changetype:modify
add: work-phone
work-phone:510/506-7000
work-phone:510/506-7001
```

- **値の削除。** `delete` 行のみ記述すると、指定した属性のすべての値が削除されます。属性行を指定した場合は、その属性から特定の値を削除できます。

```
delete: attribute name
[attribute name: value1]
```

例：

```
dn:cn=Barbara Fritchey,ou=Sales,o=Oracle,c=US
changetype:delete
delete: home-fax
```

- **値の置換。**このオプションを使用すると、新しく指定した設定で、属性の値をすべて置換できます。

```
replace:attribute name
[attribute name:value1 ...]
```

replace に属性を指定しない場合、ディレクトリは空のセットを追加します。次に、ディレクトリはその空のセットを削除要求と解釈し、エントリから属性を削除することによって対応します。この方法は、存在するかどうかわからない属性を削除する場合に便利です。

例：

```
dn:cn=Barbara Fritchey,ou=Sales,o=Oracle,c=US
changetype:modify
replace: work-phone
work-phone:510/506-7002
```

**\* changetype:delete**

この変更タイプは、エントリを削除するときに使用します。第 1 行でエントリを指定し、第 2 行で changetype に delete を指定しているため、それ以上の入力はありません。

例：

```
dn:cn=Barbara Fritchey,ou=Sales,o=Oracle,c=US
changetype:delete
```

**\* changetype:modrdn**

変更タイプに続く行に、次の形式で新規の相対識別名（RDN）を指定します。

```
newrdn: RDN
```

例：

```
dn:cn=Barbara Fritchey,ou=Sales,o=Oracle,c=US
changetype:modrdn
newrdn: cn=Barbara Fritchey-Blomberg
```

ldapmodifymt 構文

ldapmodifymt コマンドライン・ツールを使用して、複数のエントリを同時に変更できます。

ldapmodifymt は、次の構文で使します。

```
ldapmodifymt -T number_of_threads [arguments] -f filename
```

filename は、4-2 ページの「LDAP データ交換フォーマット (LDIF) 構文」で説明する仕様に従って作成された、LDIF ファイルの名前です。

**関連項目：** ldapmodifymt で使用されるその他の形式化仕様は、4-12 ページの「ldapmodify 構文」を参照してください。

例：

```
ldapmodifymt -T 5 -h node1 -p 3000 -f myentries.ldif
```

オプションの引数	説明
-a	エントリが追加対象で、入力ファイルが LDIF 形式であることを示します。(ldapadd を実行している場合、このフラグは必要ありません。)
-b	データ・ファイルにバイナリ・ファイル名が含まれていることを指定します。バイナリ・ファイル名はスラッシュで始まります。
-c	エラーが発生しても処理を継続する場合に指定します。エラーはレポートされます。(このオプションを使用しない場合、エラーが発生すると ldapmodify は停止します。)
-D binddn	ディレクトリに対して認証するとき、binddn に指定されているエントリとして認証することを指定します。この引数は、-w password オプションとともに使用します。
-E "character_set"	ネイティブ・キャラクタ・セット・エンコーディングを指定します。『Oracle Internet Directory 管理者ガイド』の NLS の章を参照してください。
-h ldaphost	デフォルトのホスト（ローカル・コンピュータ）ではなく、ldaphost に接続します。ldaphost には、コンピュータ名または IP アドレスを指定します。
-n	操作を実際には実行せずに、予測結果を示します。
-p ldapport	TCP ポート ldapport 上のディレクトリに接続します。このオプションを指定しない場合は、デフォルト・ポート（389）に接続されます。
-P wallet_password	Wallet のパスワードを指定します（サーバー、またはクライアントとサーバーの SSL 接続の場合は必須）。

オプションの引数	説明
-T	エントリを同時に処理するスレッドの数を設定します。
-U <i>SSLAAuth</i>	SSL 認証モードを指定します。 <ul style="list-style-type: none"> <li>■ 1: SSL 認証なし</li> <li>■ 2: サーバー認証</li> <li>■ 3: クライアントとサーバーの認証</li> </ul>
-v	冗長モードを指定します。
-w <i>password</i>	デフォルトの非認証の NULL バインドをオーバーライドします。認証を強制するには、このオプションを -D オプションとともに使用します。
-W <i>wallet_location</i>	Wallet の位置を指定します（サーバー、またはクライアントとサーバーの SSL 接続の場合は必須）。

## ldapsearch 構文

ldapsearch コマンドライン・ツールを使用して、ディレクトリ内の特定のエントリを検索して取り出すことができます。

ldapsearch は、次の構文で使します。

```
ldapsearch [arguments] filter [attributes]
```

フィルタの書式は RFC-2254 に準拠している必要があります。この規格の詳細は、次の Web サイトを検索してください。<http://www.ietf.org/rfc/rfc2254.txt>

属性は空白で区切ります。属性を何も入力しないと、すべての属性が取り出されます。

必須の引数	説明
-b <i>basedn</i>	検索するベース識別名（DN）を指定します。
-s <i>scope</i>	検索有効範囲を指定します。base、one または sub。

オプションの引数	説明
-A	属性名のみ取り出します（値は取り出しません）。
-a <i>deref</i>	別名参照解除を指定します。never、always、search または find。
-B	非 ASCII 値を出力します。

オプションの引数	説明
-D <i>binddn</i>	ディレクトリに対して認証するときに、 <i>binddn</i> に指定されているエントリとして認証することを指定します。この引数は、 <i>-w password</i> オプションとともに使用します。
-d <i>debug level</i>	デバッグ・レベルを、指定したレベルに設定します（『Oracle Internet Directory 管理者ガイド』のディレクトリ・サーバーの管理の章を参照してください）。
-E " <i>character_set</i> "	ネイティブ・キャラクタ・セット・エンコーディングを指定します。『Oracle Internet Directory 管理者ガイド』の NLS の章を参照してください。
-f <i>file</i>	<i>file</i> にリストされている検索順を実行します。
-F <i>sep</i>	属性名と値の間に、 <i>=</i> ではなく <i>sep</i> を出力します。
-h <i>ldaphost</i>	デフォルトのホスト（ローカル・コンピュータ）ではなく、 <i>ldaphost</i> に接続します。 <i>ldaphost</i> には、コンピュータ名または IP アドレスを指定します。
-L	エントリを LDIF 形式で出力します（引数 <i>B</i> の内容も含まれます）。
-l <i>timelimit</i>	<i>ldapsearch</i> コマンドが完了するまでの最大待機時間（秒）を指定します。
-n	検索を実際には実行せずに、予測結果を示します。
-p <i>ldapport</i>	TCP ポート <i>ldapport</i> 上のディレクトリに接続します。このオプションを指定しない場合は、デフォルト・ポート（389）に接続されます。
-P <i>wallet_password</i>	Wallet のパスワードを指定します（サーバー、またはクライアントとサーバーの SSL 接続の場合は必須）。
-S <i>attr</i>	検索結果を属性 <i>attr</i> でソートします。
-t	/tmp のファイルに書き込みます。
-u	わかりやすいエントリ名で出力します。
-U <i>SSLAuth</i>	SSL 認証モードを指定します。 <ul style="list-style-type: none"> <li>■ 1: SSL 認証なし</li> <li>■ 2: サーバー認証</li> <li>■ 3: クライアントとサーバーの認証</li> </ul>
-v	冗長モードを指定します。
-w <i>bindpassword</i>	バインド・パスワードを指定します（簡易認証の場合）。

オプションの引数	説明
<code>-W wallet_location</code>	Wallet の位置を指定します（サーバー、またはクライアントとサーバーの SSL 接続の場合は必須）。
<code>-z sizelimit</code>	エントリの最大検索数を指定します。

### ldapsearch フィルタの例

検索コマンドの作成方法を理解するには、次の例を参考にしてください。

**例 1: ベース・オブジェクト検索** 次の例は、ルートからディレクトリ上のベース・レベルの検索を実行します。

```
ldapsearch -p 389 -h myhost -b "" -s base -v "objectclass=*"
```

- `-b` オプションは、検索するベース識別名（DN）を指定します。この場合はルートです。
- `-s` オプションは、検索がベース検索（base）か、1 レベルの検索（one）か、サブツリー検索（sub）かを指定します。
- `"objectclass=*"` は、検索時のフィルタを指定します。

**例 2: 1 レベルの検索** 次の例は、識別名（DN）`"ou=HR, ou=Americas, o=IMC, c=US"` から開始する 1 レベルの検索を実行します。

```
ldapsearch -p 389 -h myhost -b "ou=HR, ou=Americas, o=IMC, c=US" -s one -v "objectclass=*"
```

**例 3: サブツリー検索** 次の例は、サブツリー検索を実行し、`"cn=Person"` で始まる DN を持つすべてのエントリを戻します。

```
ldapsearch -p 389 -h myhost -b "c=US" -s sub -v "cn=Person"
```

**例 4: サイズ制限を使用した検索** 次の例では、一致するエントリが 3 つ以上ある場合でも、実際には 2 つのみが取り出されます。

```
ldapsearch -h myhost -p 389 -z 2 -b "ou=Benefits,ou=HR,ou=Americas,o=IMC,c=US" -s one "objectclass=*"
```

**例 5: 指定した属性および属性オプションでの検索** 次の例では、一致するエントリの DN 属性値のみを戻します。

```
ldapsearch -p 389 -h myhost -b "c=US" -s sub -v "objectclass=*" dn
```

次の例では、surname (sn)、description (description) 属性値とともに、識別名 (dn) 属性値のみが取り出されます。

```
ldapsearch -p 389 -h myhost -b "c=US" -s sub -v "cn=Person*" dn sn description
```

次の例では、言語コード属性オプションを指定するオプションがある一般名 (cn) 属性を持ったエントリを取り出します。この例では、一般名がフランス語で、R の文字から始まるエントリを取り出します。

```
ldapsearch -p 389 -h myhost -b "c=US" -s sub "cn;lang-fr=R"
```

John のエントリには、cn;lang-it 言語コード属性オプションの値が設定されていないものとします。その場合、次の例を実行すると失敗します。

```
ldapsearch -p 389 -h myhost -b "c=us" -s sub "cn;lang-it=John"
```

**例 6: すべてのユーザー属性および指定した操作属性の検索** 次の例では、すべてのユーザー属性と、createtimestamp 操作属性および orclguid 操作属性を取り出します。

```
ldapsearch -p 389 -h myhost -b "ou=Benefits,ou=HR,ou=Americas,o=IMC,c=US" -s sub "cn=Person*" * createtimestamp orclguid
```

次の例では、Anne Smith が修正したエントリを取り出します。

```
ldapsearch -h sun1 -b "" "(&(objectclass=*)(modifiersname=cn=Anne Smith))"
```

次の例では、2000 年 4 月 1 日から 2000 年 4 月 6 日の間に修正されたエントリを取り出します。

```
ldapsearch -h sun1 -b "" "(&(objectclass=*)(modifytimestamp>=20000401000000)(modifytimestamp<= 20000406235959))"
```

---

---

**注意：** modifiersname および modifytimestamp は索引付き属性ではないので、catalog.sh を使用してこれら 2 種類の属性に索引付けをします。そして、Oracle Internet Directory サーバーを再起動してから、前述の 2 つの ldapsearch コマンドを発行します。

---

---

**その他の例：** 次の各例は、ホスト sun1 のポート 389 で、識別名 (DN) "ou=hr,o=acme,c=us" から開始してサブツリー全体を検索します。

次の例は、objectclass 属性の値を持つエントリをすべて検索します。

```
ldapsearch -p 389 -h sun1 -b "ou=hr, o=acme, c=us" -s subtree "objectclass=*
```

次の例は、objectclass 属性の値が orcle で始まるエントリをすべて検索します。

```
ldapsearch -p 389 -h sun1 -b "ou=hr, o=acme, c=us" -s subtree "objectclass=orcle"
```

次の例は、objectclass 属性が orcle で始まり、cn が foo で始まるエントリを検索します。

```
ldapsearch -p 389 -h sun1 -b "ou=hr, o=acme, c=us" -s subtree
"(&(objectclass=orcle*)(cn=foo*))"
```

次の例は、一般名 (cn) が foo 以外のエントリを検索します。

```
ldapsearch -p 389 -h sun1 -b "ou=hr, o=acme, c=us" -s subtree "(!(cn=foo))"
```

次の例は、cn が foo で始まるか、または sn が bar で始まるエントリを検索します。

```
ldapsearch -p 389 -h sun1 -b "ou=hr, o=acme, c=us" -s subtree
"(|(cn=foo*)(sn=bar*))"
```

次の例は、employeenumber が 10,000 以下のエントリを検索します。

```
ldapsearch -p 389 -h sun1 -b "ou=hr, o=acme, c=us" -s subtree
"employeenumber<=10000"
```

## カタログ管理ツールの構文

Oracle Internet Directory は、索引を使用して属性を検索できるようにしています。Oracle Internet Directory のインストール時に、エントリ cn=catalogs に、検索で使用できる属性がリストされます。等価の一致規則を持つ属性のみが索引付けできます。

その他の属性を検索フィルタで使用する場合は、使用する属性をカタログ・エントリに追加する必要があります。追加は、Oracle Directory Manager を使用して属性を作成するときに可能です。しかし、属性がすでに存在している場合は、カタログ管理ツールを使用しないと、その属性に索引付けできません。

カタログ管理ツールを実行する前に、LANG 変数の設定を解除してください。カタログ管理ツールの実行終了後、LANG 変数を元の値に設定してください。

LANG の設定を解除する方法は、次のとおりです。

- Korn シェルを使用している場合

```
UNSET LANG
```

- C シェルを使用している場合

```
UNSETENV LANG
```

カタログ管理ツールは、次の構文で使します。

```
catalog.sh -connect net_service_name {add|delete} {-attr attr_name|-file filename}
```

必須の引数	説明
- connect <i>net_service_name</i>	ディレクトリ・データベースに接続するためのネット・サービス名を指定します。
	関連項目：『Oracle8i Net8 管理者ガイド』

オプションの引数	説明
- add -attr <i>attr_name</i>	指定した属性を索引付けします。
- delete -attr <i>attr_name</i>	指定した属性から索引を削除します。
- add -file <i>filename</i>	指定したファイル内の属性（1 行に 1 つずつ）を索引付けします。
-delete -file <i>filename</i>	指定したファイル内の属性から索引を削除します。

catalog.sh コマンドを入力すると、次のメッセージが表示されます。

```
This tool can only be executed if you know the OiD user password.  
Enter OiD password:
```

正しいパスワードを入力すると、コマンドが実行されます。パスワードに誤りがあると、次のメッセージが表示されます。

```
Cannot execute this tool
```

カタログ管理ツールの実行終了後、LANG 変数を元の値に設定してください。

LANG を設定する方法は、次のとおりです。

- Korn シェルを使用している場合  
SET LANG=*appropriate\_language*; EXPORT LANG
- C シェルを使用している場合  
SETENV LANG *appropriate\_language*

カタログ管理ツールの実行後にその変更内容を有効にするには、Oracle Internet Directory サーバーを停止して再起動してください。

**関連項目：** ディレクトリ・サーバーの起動および再起動の方法は、『Oracle Internet Directory 管理者ガイド』の事前に行う作業の章を参照してください。

---

# 索引

## A

---

add.log, 4-6

## C

---

### C API

- SSL モードでの使用方法, 2-6
- サンプル検索ツール, 2-8
- 使用例, 2-6
- 非 SSL モードでの使用方法, 2-7

C API の使用例, 2-6

catldap.sql, 3-13

Chadwick, David, iv

changetype

- add, 4-14
- delete, 4-15
- modify, 4-14
- modrdn, 4-15

## D

---

DBMS\_LDAP.init ファンクション, 3-20

DBMS\_LDAP パッケージ, 3-2, 3-13

## H

---

Hodges, Jeff, iv

Howes, Tim and Mark Smith, iv

## J

---

Java, 1-2

JNDI, 1-2

JPEG イメージ、ldapadd を使用した追加, 4-6

## K

---

Kerberos 認証, 4-5, 4-7, 4-10

Kosiur, Dave, iv

## L

---

### LDAP

検索フィルタ、IETF 準拠, 4-17

ldap\_init\_SSL コール, 2-3

ldapadd, 4-4

JPEG イメージの追加, 4-6

エントリの追加, 4-4

構文, 4-4

ldapaddmt, 4-6

構文, 4-6

複数エントリを同時に追加, 4-6

ログ, 4-6

ldapbind, 4-7

構文, 4-7

ldapcompare, 4-8

構文, 4-8

ldapdelete, 4-10

エントリの削除, 4-10

構文, 4-10

ldapmoddn, 4-11

構文, 4-11

ldapmodify, 4-12

LDIF ファイル, 4-4, 4-6, 4-12, 4-16

エントリの削除, 4-15

グループ・エントリの作成, 4-14

構文, 4-12

属性値の置換, 4-15

複数値の属性への値の追加, 4-14

変更の種類, 4-14

- ldapmodifymt, 4-16
  - 構文, 4-16
  - 使用方法, 4-16
  - マルチスレッド処理, 4-17
- ldapsearch, 2-8, 4-17
  - 構文, 4-17
  - フィルタ, 4-19
- LDAP サーバー、サード・パーティ, 2-20
- LDAP セッション
  - 開始と終了, 2-4
- LDAP 操作、実行, 2-4
- LDAP データ交換フォーマット (LDIF), 4-2
- LDAP バージョン 2 C API, 2-2
- LDIF
  - 形式化規則, 4-3
  - 形式化の注意事項, 4-3
  - 構文, 4-2
  - 使用方法, 4-2
  - ファイル、ldapmodify コマンド, 4-4, 4-6, 4-12, 4-16

## O

---

- Oracle Directory Manager, 1-2
  - 属性の型のリスト, 4-3
- Oracle Directory Replication Server, 1-2
- Oracle Internet Directory
  - コンポーネント, 1-2
- Oracle Internet Directory SDK のコンポーネント, 1-2
- Oracle Internet Directory サーバー, 1-2
- Oracle Internet Directory でサポートされるオペレーティング・システム, 1-3
- Oracle SSL 関連ライブラリ, 2-21
- Oracle SSL 機能拡張, 2-2
- Oracle SSL コール・インタフェース, 2-2
- Oracle Wallet, 2-3
- Oracle Wallet Manager, 2-3
  - Wallet を作成するために必要, 2-21
- Oracle システム・ライブラリ, 2-21

## P

---

- PL/SQL API, 3-1
  - C API の一部を含む, 3-2
  - アプリケーションの作成, 3-13
  - 依存性と制限事項, 3-13
  - 検索方法, 3-9

- サブプログラム, 3-19
- サブプログラムの概要, 3-14
- サンプル, 3-2
- データ型の概要, 3-18
- データベース・トリガーからの使用, 3-2
- データベースへのロード, 3-13
- 例外の概要, 3-16
- PL/SQL の使用例, 3-2
- PL/SQL のリファレンス, 3-13
- PL/SQL ファンクション
  - add\_s, 3-62
  - bind\_s, 3-23
  - compare\_s, 3-27
  - count\_entries, 3-37
  - count\_values, 3-65
  - count\_values\_len, 3-66
  - create\_mod\_array, 3-54
  - delete\_s, 3-49
  - err2string, 3-53
  - explode\_dn, 3-69
  - first\_attribute, 3-39
  - first\_entry, 3-33
  - get\_dn, 3-43
  - get\_values, 3-45
  - get\_values\_len, 3-47
  - init, 3-19
  - modify\_s, 3-60
  - modrdn2\_s, 3-51
  - next\_attribute, 3-41
  - next\_entry, 3-35
  - open\_ssl, 3-71
  - rename\_s, 3-67
  - search\_s, 3-29
  - search\_st, 3-31
  - simple\_bind\_s, 3-21
  - unbind\_s, 3-25
- PL/SQL プロシージャ
  - free\_mod\_array, 3-64
  - populate\_mod\_array (バイナリ・バージョン), 3-58
  - populate\_mod\_array (文字列バージョン), 3-56

## R

---

- Radicati, Sara, v
- RFC 1823, 2-21

## S

---

SDK コンポーネント, 1-2  
SQL\*Plus, 3-13  
SSL  
    Oracle の機能拡張, 2-2  
        暗号化と復号化, 2-2  
    Wallet, 2-3  
    インタフェース・コール, 2-2  
    使用可能, 4-5, 4-7, 4-8, 4-13, 4-17  
    認証モード, 2-2  
    ハンドシェイク, 2-3  
SSL をサポートする Oracle の機能拡張, 2-2

## T

---

TCP/IP ソケット・ライブラリ, 2-21

## W

---

Wallet, SSL, 2-3

## あ

---

アプリケーションの作成  
    PL/SQL API を使用した, 3-13

## い

---

依存性と制限事項  
    C API, 2-20  
    PL/SQL API, 3-13  
インタフェース・コール, SSL, 2-2

## え

---

エラー, C API での操作, 2-5  
エントリ  
    検索, ldapsearch を使用, 4-17  
    削除  
        ldapdelete を使用, 4-10  
        ldapmodify を使用, 4-15  
    追加  
        ldapaddmt を使用, 4-6  
        ldapadd を使用, 4-4  
    変更  
        同時, ldapmodifymt を使用, 4-16

## お

---

オブジェクト・クラス  
    LDIF ファイル, 4-2  
    追加  
        同時, ldapaddmt を使用, 4-6

## か

---

管理ツール  
    ldapadd, 4-4  
    ldapaddmt, 4-6  
    ldapbind, 4-7  
    ldapcompare, 4-8  
    ldapdelete, 4-10  
    ldapmoddn, 4-11  
    ldapmodify, 4-12  
    ldapmodifymt, 4-16  
    ldapsearch, 4-17

## き

---

規則, LDIF, 4-3

## く

---

クライアントとサーバーの認証, SSL, 2-2  
グループ・エントリ  
    作成, ldapmodify を使用, 4-14

## け

---

検索結果、取得, 2-5  
検索フィルタ  
    IETF 準拠, 4-17  
    ldapsearch, 4-19  
検索, ldapsearch を使用, 4-17

## こ

---

構文  
    ldapadd, 4-4  
    ldapaddmt, 4-6  
    ldapbind, 4-7  
    ldapcompare, 4-8  
    ldapdelete, 4-10  
    ldapmoddn, 4-11

- ldapmodify, 4-12
- ldapmodifymt, 4-16
- ldapsearch, 4-17
- LDIF, 4-2
- カタログ管理ツール, 4-21
- コマンドライン・ツール, 4-4
- コマンドライン・ツール
  - ldapadd, 4-4
  - ldapaddmt, 4-6
  - ldapbind, 4-7
  - ldapcompare, 4-8
  - ldapdelete, 4-10
  - ldapmoddn, 4-11
  - ldapmodify, 4-12
  - ldapmodifymt, 4-16
  - ldapsearch, 4-17
  - 構文, 4-4

## さ

---

- サーバー SSL 認証, 2-2
- 削除
  - エントリ
    - ldapdelete を使用, 4-10
    - ldapmodify を使用, 4-15
  - オブジェクト、コマンドライン・ツールを使用, 4-10, 4-12
- 属性
  - ldapmodify を使用, 4-15
  - 属性からの値の削除、ldapmodify を使用, 4-14

## し

---

- 識別名
  - C API での操作, 2-5
  - LDIF ファイル, 4-2

## そ

---

- 相対識別名 (RDN)
  - 変更
    - ldapmodify を使用, 4-15
- 属性
  - LDIF ファイル, 4-2
  - 削除, 4-15
    - 値、ldapmodify を使用, 4-14

- 追加
  - ldapadd を使用, 4-4
  - 既存のエントリ, 4-4
  - 同時、ldapaddmt を使用, 4-6
- 属性値の置換、ldapmodify を使用, 4-15

## つ

---

- 追加
  - エントリ
    - ldapaddmt を使用, 4-6
    - ldapadd を使用, 4-4
    - 同時, 4-6
  - 既存のエントリへの属性の追加, 4-4

## に

---

- 認証
  - Kerberos, 4-5, 4-7, 4-10
  - SSL, 2-2, 4-5, 4-7, 4-8, 4-13, 4-17
    - クライアントとサーバー, 2-2
  - サーバー, 2-2
  - 認証なし, 2-2
  - モード、SSL, 2-2

## は

---

- パフォーマンス
  - 複数のスレッドの使用, 4-6
- バルク・ツール, 1-2

## ふ

---

- フィルタ
  - IETF 準拠, 4-17
  - ldapsearch, 4-19
- 複数値の属性
  - 値の追加、ldapmodify を使用, 4-14
- 複数のスレッド, 4-17
  - ldapaddmt, 4-6
  - 数の増加, 4-6

## へ

---

変更

    エントリ

        ldapmodify を使用, 4-12

        同時、ldapmodifymt を使用, 4-16

変更の種類、ldapmodify 入力ファイル, 4-14

## ま

---

マルチスレッド・コマンドライン・ツール

    ldapaddmt, 4-6

    ldapmodifymt, 4-17

## め

---

メモリー、C API での解放, 2-5

