OpenLDAP データの SQL 格納

OpenLDAP は、デフォルトではデータをバークレー DB ファイルに格納するが、データを SQL データベースに格納することができるが、SQL データベースを使うためには OpenLDAP のデータの格納先に sql を指定して、ODBC 経由でデータベースサーバにアクセスする必要がある。ここでは、ODBC ドライバを設定して、OpenLDAP のデータを PostgreSQL に格納する方法を簡単なサンプルとともに紹介する。OpenLDAP のサービスを提供するプログラムは slapd で、その SQL インターフェースということで、OpenLDAP で SQL データベースにデータを格納するための設定は SLAPD-SQL とも呼ばれる。

まず、最初にこの SLAPD-SQL の OpenLDAP のサーバーの設定について説明し、次に、PostgreSQL データベースサーバに接続するための設定について説明する。SLAPD-SQL では、ODBC ドライバを介してデータベースサーバに接続する。具体的な設定例の説明にあたっては、OpenLDAP の配布に含まれるサンプルを利用する。

SLAPD-SQL

OpenLDAP の設定は slapd.conf に行う。SQL データベースに格納するためには、"database sql " を指定する。この設定は、俗に SLAPD-SQL と記述され、オンラインマニュアルも存在する（たとえば、

http://techpubs.sgi.com/library/tpl/cgi-bin/getdoc.cgi?coll=0650&db=man&fname=/usr/share/catman/p_man/cat5/openldap/slapd-sql.z ）。

SLAPD-SQL で使う際の slapd.conf 内のデータベース定義部分の設定例を次に示す。

```
########################################################################
# sql database definitions
########################################################################

database     sql
suffix       "dc=example,dc=com"
rootdn       "cn=root,dc=example,dc=com"
rootpw       secret
dbname       PgLDAP
dbuser       ldap
dbpasswd     pass
insentry_query "insert into ldap_entries (id,dn,oc_map_id,parent,keyval) \
             values ((select max(id)+1 from ldap_entries),?,?,?,?)"
upper_func         "upper"
strcast_func       "text"
concat_pattern     "?||?"
has_ldapinfo_dn_ru  no

lastmod      off
```

上記の設定の主な意味を説明する。"dbname PgLDAP" は ODBC ドライバのエントリーを示す。 "dbuser ldap"と"dbpasswd pass"はデータベースにアクセスするためのユーザ名とパスワードを指定する。

ODBC ドライバ

ここでは、SLAPD-SQL の設定で、ODBC ドライバとして unixODBC を利用する例を示す。unixODBC は、*http://www.unixodbc.org/*にあり、Linux でも使える。ここでは、unixODBC ドライバがインストールされていることを前提に説明を行う。

unixODBC ドライバの設定は、ドライバマネージャの設定を odbcinst.ini ファイルに、ドライバの設定を[.]odbc.ini ファイルに行う（[.]が付くのは Linux の各ユーザ毎にホームディレクトリに設定ファイルを設置する場合）。

1. odbcinst.ini

ODBC ドライバマネージャの設定は、unixODBC のデフォルトでは/usr/etc/odbcinst.ini に行う。このファイルでは ODBC ドライバ名とそのライブラリを定義する。たとえば、次のように記述する。

```
[PostgreSQL]
Description = PostgreSQL driver for Linux & Win32
Driver = /usr/lib/libodbcpsql.so
Setup = /usr/lib/libodbcpsqlS.so
FileUsage = 1
UsageCount = 1
```

あるいは、上記内容のテンプレートファイルに作成しておけば、次のように odbcinst コマンドでシステム共通の所定の場所に設置することができる。

```
# odbcinst -i -d -f odbcinst.ini.template
（デフォルトの unixODBC では/usr/etc/odbcinst.ini）
```

上記のドライバマネージャの設定例では、エントリのドライバ名は"PostgreSQL"になっている。このエントリーが指定された際に"Driver = /usr/lib/libodbcpsql.so"に指定されたランタイムライブラリを参照するようになる。

2. [.]odbc.ini

ODBC ドライバエントリーの記述例として、ここでは PostgreSQL ODBC ドライバを設定する例を示す。

```
[PgLDAP]
Description = PostgreSQL LDAP DBC
Driver = PostgreSQL
Trace = Yes
TraceFile = odbc-pgldap.log
Database = pg_ldap
Servername = localhost
Username = ldap
Password = pass
Port = 5432
Protocol = 7.2.3
ReadOnly = No
RowVersioning = No
ShowSystemTables = No
```

```
ShowOidColumn = No
FakeOidIndex = No
ConnSettings =
```

あるいは、上記内容のテンプレートファイルに作成して、次のように odbcinst コマンドで所定の場所に設置することができる。

```
$ odbcinst -i -s -f _odbc.ini.template
```
（デフォルトは、$HOME/.odbc.ini）

このドライバの設定ファイルでは、エントリ名は"PgLDAP"で、これが先に例示した slapd.conf の中では"dbname"に指定されている。"Driver = PostgreSQL"で実際の ODBC ドライバのエントリを指定している。

3. アクセステスト

unixODBC に含まれる isql コマンドは unixODBC ドライバのコマンド行インターフェースプログラムで、指定したドライバエントリへの接続を行ない、SQL 命令を発行できる。この isql プログラムを使って簡単にデータベースへの接続テストが行える。

```
$ isql PgLDAP
+---------------------------------------+
| Connected!
|
| sql-statement
| help [tablename]
| quit
|
+---------------------------------------+
SQL> quit
$
```

4.psqlodbc ドライバ

ODBC ドライバとして、PostgreSQL の最新の psqlodbc ドライバを利用する場合は、

http://www.postgresql.org/ftp/odbc/versions/src/

からダウンロードしてインストールを行う。psqlodbc をインストールすることで、UNICODE 対応の PostgreSQL 用 ODBC ドライバをインストールすることが可能である。

この psqlodbc の UNICODE 対応ドライバのファイル名は psqlodbcw.so で、次のように odbcinst.ini の"Driver"行に指定する。

```
Driver = /opt/pgsql/lib/psqlodbcw.so
```

slapd-sql テストデータベースの作成

ここでは、OpenLDAP の配布に含まれる slapd-sql のサンプルデータベースのテーブルを作成する手順を説明する。テーブルの作成にあたり、まず、データベースとユーザを作成し、次に、LDAP のための SQL バックエンドを作成する。そして、メタデータを作成し、テストデータを登録する。テーブルに slapd からのアクセス権を付与する。

1. データベースとユーザを作成

PostgreSQL に slapd からアクセスするためのユーザを作成し、データベースを作成する。

```
$ createuser --no-createdb --no-adduser --password ldap
(プロンプトに従ってパスワードを入力)
$ createdb -O ldap -E UTF-8 pg_ldap
```

2. LDAP のための SQL バックエンドを作成

OpenLDAP の SQL バックエンドとして稼働させるために、データベース構造と情報の登録が必要である。OpenLDAP の配布に(openldap-2.3.x/servers/slapd/back-sql/rdbms_depend/pgsql/ディレクトリ)雛型が含まれているので、それを使って作成する。

```
$ psql -U ldap pg_ldap < backsql_create.sql
```

ここで登録したテーブルは OpenLDAP のオブジェクト間のすべてのリンクを維持するために使われる。これが、LDAP メタ構造(meta-structure)である。

3. テスト用のデータベーススキーマの作成

テスト用の LDAP オブジェクトを再現するテーブルのためのスキーマを作成する。

```
$ psql -U ldap -d pg_ldap < testdb_create.sql
```

ここで作成したテーブルはテスト用の LDAP ディレクトリオブジェクトと属性(attributes)をつくるために使われる。

4. メタデータの作成

ここでは、SQL バックエンドと格納されたオブジェクトの間のリンクをテストデータベースに生成する。これらのメタ情報は LDAP クエリを SQL クエリに変換するために使われる。また、 SQL バックエンドと格納されたオブジェクトの間のリンクを生成したり、全属性値を格納したりするために使われるメタデータの定義と SQL 関数のすべてをテストデータベースに生成する。

```
$ psql -U ldap -d pg_ldap < testdb_metadata.sql
```

5. テストのためのデータを挿入

ここで、テスト用のデータをデータベースに挿入する。

```
$ psql -U ldap -d pg_ldap < testdb_data.sql
```

6. データベースオブジェクトのアクセス権付与

データベースのオーナーがアクセスするユーザと異なる場合は、データベースへのアクセス権を与える必要がある。

```
$ psql -U ldap -d pg_ldap -c \
      "GRANT ALL ON
            ldap_attr_mappings,
            ldap_entries,
            ldap_entry_objclasses,
            ldap_oc_mappings,
            ldap_referrals
      TO ldap;"
$ psql -U ldap -d pg_ldap -c \
      "GRANT ALL ON
            ldap_attr_mappings_id_seq,
            ldap_entries_id_seq,
            ldap_oc_mappings_id_seq
      TO ldap;"
$ psql -U ldap -d pg_ldap -c \
      "GRANT ALL ON
            authors_docs, documents, institutes, persons, phones
      TO ldap;"
$ psql -U ldap -d pg_ldap -c \
      "GRANT ALL ON
            documents_id_seq, institutes_id_seq,
            persons_id_seq, phones_id_seq
      TO ldap;"
```

SLAPD-SQL の実行とテスト

ここでは、設定の終了した spapd を実行し、LDAP クライアントあるいは LDAP ブラウザからアクセスをして LDAP サーバの動作を確認する。OpenLDAP サーバの起動は slapd を実行する。

```
# /usr/libexec/slapd
```

テストではデバグを出力させながら実行させるために、-d オプションを指定する。

```
# /usr/libexec/slapd -d 5
```

OpenLDAP の slapd にアクセスするクライアントプログラムには ldapsearch, ldapadd, ldapmodify, ldapdelete などがある。これらのコマンドを使って SLAPD-SQL で実際にサービスが提供されているテストを行う。

1．検索テスト

ldapsearch コマンドで(objectClass=*)を指定して、すべてのオブジェクトを検索する。

```
$ ldapsearch -x -b "dc=example,dc=com" "(objectClass=*)"
# extended LDIF
#
# LDAPv3
# base with scope subtree
# filter: (objectClass=*)
# requesting: ALL
#

# book1, example.com
dn: documentTitle=book1,dc=example,dc=com
objectClass: document
description: abstract1
documentTitle: book1
documentAuthor: cn=Mitya Kovalev,dc=example,dc=com
documentAuthor: cn=Torvlobnor Puzdoy,dc=example,dc=com
documentIdentifier: document 1

# book2, example.com
dn: documentTitle=book2,dc=example,dc=com
objectClass: document
description: abstract2
documentTitle: book2
documentAuthor: cn=Mitya Kovalev,dc=example,dc=com
documentIdentifier: document 2

# search reference
ref: ldap://localhost:9012/dc=example,dc=com??sub

# example.com
dn: dc=example,dc=com
objectClass: organization
```

```
objectClass: dcObject
o: Example
dc: example

# Mitya Kovalev, example.com
dn: cn=Mitya Kovalev,dc=example,dc=com
objectClass: inetOrgPerson
cn: Mitya Kovalev
sn: Kovalev
seeAlso: documentTitle=book1,dc=example,dc=com
seeAlso: documentTitle=book2,dc=example,dc=com
givenName: Mitya
userPassword:: bWl0
telephoneNumber: 222-3234
telephoneNumber: 332-2334

# Torvlobnor Puzdoy, example.com
dn: cn=Torvlobnor Puzdoy,dc=example,dc=com
objectClass: inetOrgPerson
cn: Torvlobnor Puzdoy
sn: Puzdoy
seeAlso: documentTitle=book1,dc=example,dc=com
givenName: Torvlobnor
telephoneNumber: 545-4563

# Akakiy Zinberstein, example.com
dn: cn=Akakiy Zinberstein,dc=example,dc=com
objectClass: inetOrgPerson
cn: Akakiy Zinberstein
sn: Zinberstein
givenName: Akakiy

# search result
search: 2
result: 0 Success

# numResponses: 8
# numEntries: 6
# numReferences: 1
```

2. エントリの作成

ldapadd コマンドを使ってエントリの追加をする。追加するエントリは、newentry.ldif ファイルに定義してある。newentry.ldif の内容は次のとおり。

```
# begin
dn: cn=User Test_Add_Entry,dc=example,dc=com
objectClass: inetOrgPerson
sn: First Test_Add_Entry user
cn: User Test_Add_Entry
# end
```

このファイルを指定し、ldapadd を実行する。

```
$ ldapadd -x -D "cn=root,dc=example,dc=com" -w secret -f newentry.ldif
adding new entry "cn=User Test_Add_Entry,dc=example,dc=com"
```

追加したエントリを検索する。

```
$ ldapsearch -x -b "dc=example,dc=com" \
            "(&(objectClass=inetOrgPerson) (cn=User*))"
# extended LDIF
#
# LDAPv3
# base with scope subtree
# filter: (&(objectClass=inetOrgPerson) (cn=User*))
# requesting: ALL
#

# User Test_Add_Entry, example.com
dn: cn=User Test_Add_Entry,dc=example,dc=com
objectClass: inetOrgPerson
cn: User Test_Add_Entry
sn: Test_Add_Entry
givenName: User

# search result
search: 2
result: 0 Success
```

3. エントリ属性の設定

ldapmodify コマンドを使ってエントリ属性を設定する。属性設定をするエントリは、setattrib .ldif ファイルに定義してある。setattrib.ldif の内容は次のとおり。

```
# begin

dn: cn=User Test_Add_Entry,dc=example,dc=com
changetype: modify
replace: sn
sn: Test_Add_Entry user

dn: cn=User Test_Add_Entry,dc=example,dc=com
changetype: modify
add: telephoneNumber
telephoneNumber: 123-4567
telephoneNumber: 765-4321

# end
```

このファイルを指定し、ldapmodify を実行する。

```
$ ldapmodify -x -D "cn=root,dc=example,dc=com" -w secret -f setattrib.ldif
modifying entry "cn=User Test_Add_Entry,dc=example,dc=com"
ldap_modify: Naming violation (64)
additional info: value of naming attribute 'cn' is not present in entry
```

4. エントリ属性の削除

ldapmodify コマンドを使ってエントリ属性を削除する。属性の削除をするエントリは、

deleteattrib .ldif ファイルに定義してある。deleteattrib.ldif の内容は次のとおり。

```
# begin

dn: cn=User Test_Add_Entry,dc=example,dc=com
changetype: modify
delete: telephoneNumber
telephoneNumber: 332-2334

# end
```

このファイルを指定して ldapmodify を実行する。

```
$ ldapmodify -x -D "cn=root,dc=example,dc=com" -w secret \
                                                -f deleteattr.ldif
modifying entry "cn=User Test_Add_Entry,dc=example,dc=com"
```

5. エントリの削除

ldapadd コマンドを使ってエントリを削除する。追加するエントリは、delentry.ldif ファイルに定義してある。delentry.ldif の内容は次のとおり。

```
# begin

dn: cn=User Test_Add_Entry,dc=example,dc=com
changetype: delete

# end
```

```
$ ldapadd -x -D "cn=root,dc=example,dc=com" -w secret -f delentry.ldif
deleting entry "cn=User Test_Add_Entry,dc=example,dc=com"
```

参考文献

http://www.samse.fr/GPL/ldap_pg/HOWTO/