wirkungsvoll
lösungsorientiert
planbar

Oracle und LDAP

Zugriff auf LDAP-Daten aus einer Oracle-DB

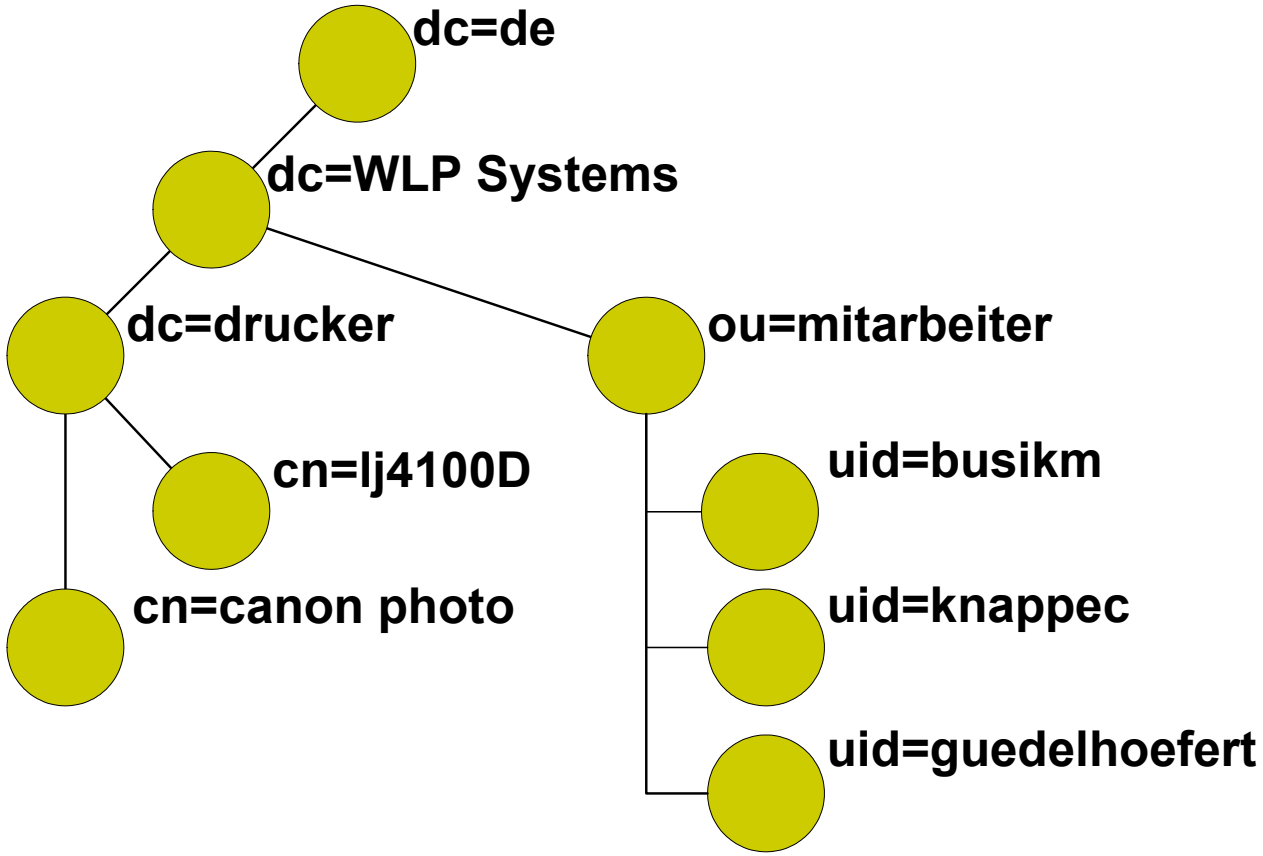
Martin Busik
busik@wlp-systems.de

Lightweight Directory Access Protocol

LDAP

Martin Busik • WLP Systems GmbH • busik@wlp-systems.de

LDAP

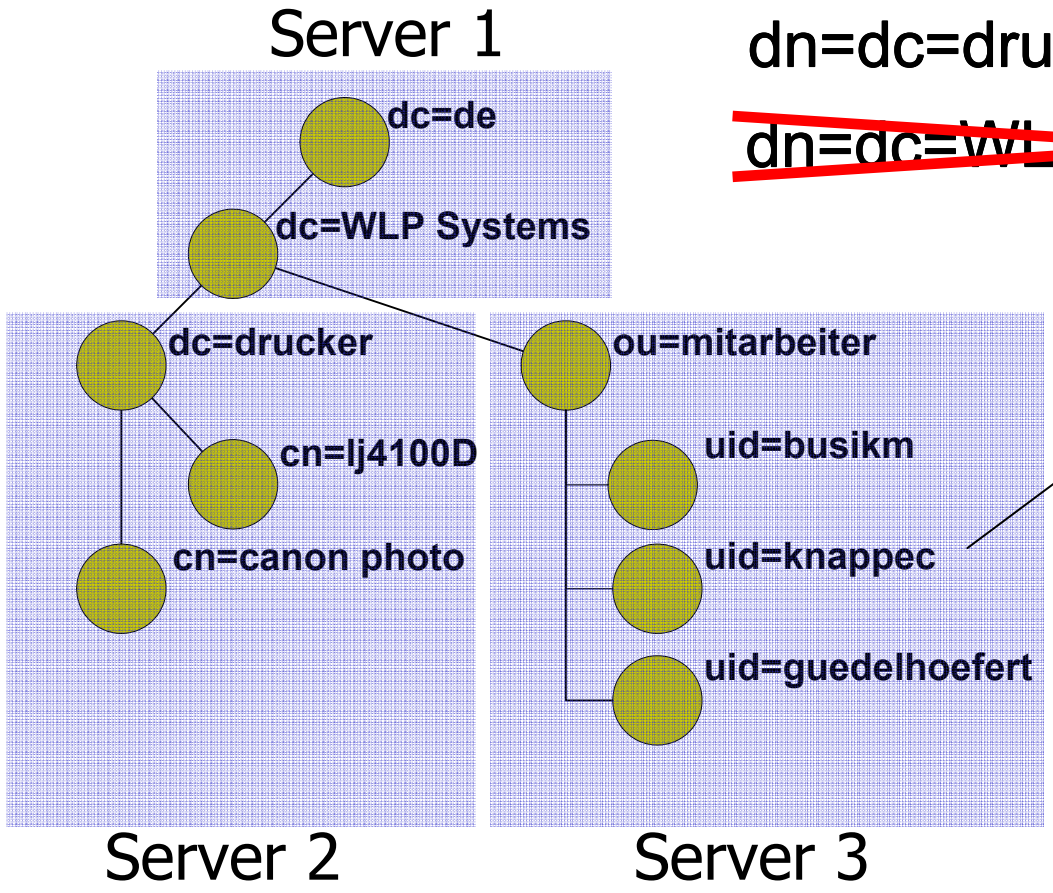

dc=de
dc=WLP Systems
dc=drucker
ou=mitarbeiter
cn=lj4100D
cn=canon photo
uid=busikm
uid=knappec
uid=guedelhoefert

SQL-Zugriff

```
select per_nachname,per_vorname,per_email
   from v_ldap_person
```

```
create view v_ldap_person as
   select * from table(ldap_person.do_select)
```

Daten in LDAP

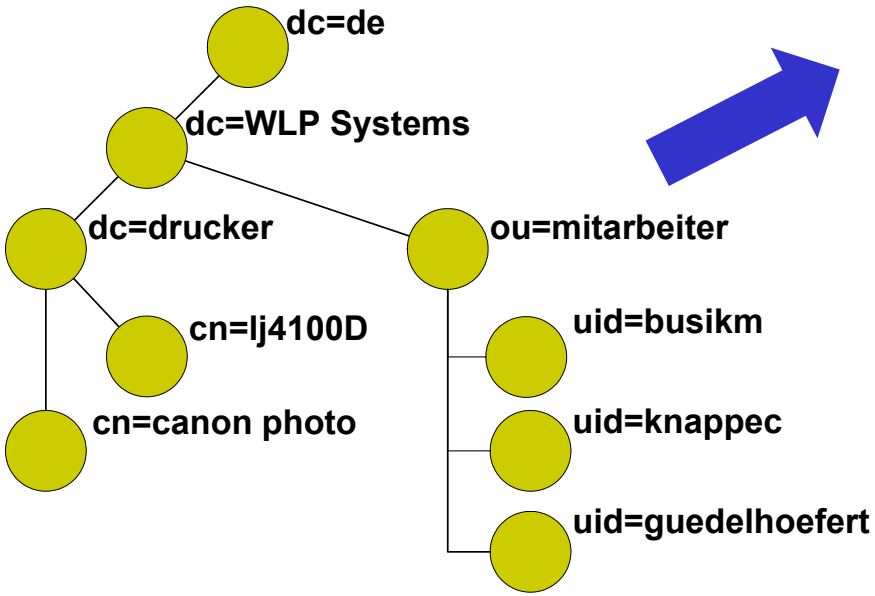


dn=dc=drucker,dc=WLP Systems,dc=de

~~dn=dc=WLP Systems,dc=drucker,dc=de~~

objectClass=
top
person
organizationalPerson
inetOrgPerson

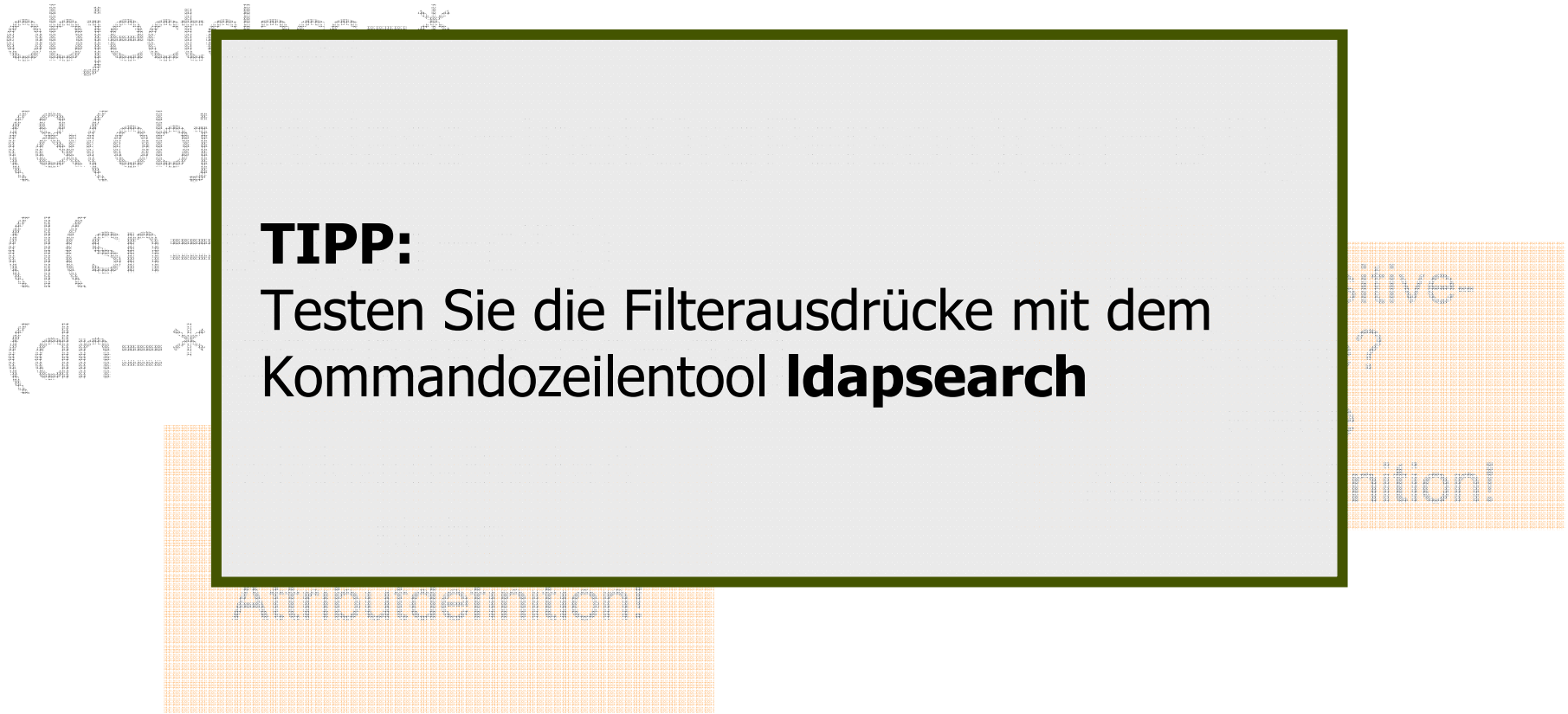
Transformation in tabellarische Struktur



nachname	vorname	email	uid
Busik	Martin	busik@	501
Knappe	Chris	knappe@	502
Eberle	Vincent	eberle@	503

Multivalue-Attribute?!

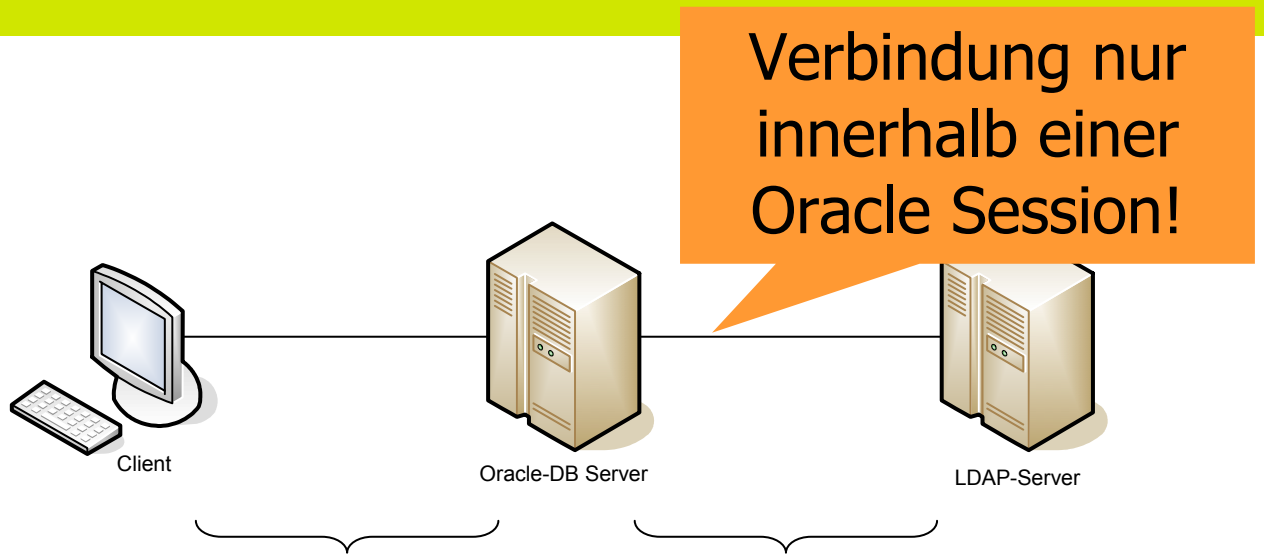
Formulierung der Filterbedingungen



TIPP:
Testen Sie die Filterausdrücke mit dem Kommandozeilentool **ldapsearch**

Martin Busik • WLP Systems GmbH • busik@wlp-systems.de

LDAP - Authentisierung



Standard-Oracle-Authentisierung

- anonyme Anmeldung
- simple bind
- simple bind über SSL
- SASL

Zugriff mittels PL/SQL - Datentypen

```
create type t_ldap_person as object (
    << einzelne Felder >> )
create type t_ldap_person_table as table of
   t_ldap_person
create function    do_select ()
   return     t_ldap_person_table
select * from table(do_select)
```

Implementierung der Function

```
create function do_select () return
   t_ldap_person_table pipelined
while loc_entry is not null loop
   << transformation >>
   pipe row(loc_person)
   end loop
return
```

Update der LDAP-Daten durch PL/SQL oder gar SQL?

procedure do_update(...)

instead of trigger

Fazit

Einsatzgebiete:

- (SQL-) Reports
- Migrationsszenarien

Alternativen:

- Batch-Jobs (ldapsearch/sql*imp/perl)
- andere Architekturansätze

Vielen Dank für Ihr Interesse

Martin Busik

WLP Systems GmbH
Alter Teichweg 25a, 22081 Hamburg

Telefon: 040 – 188 81 17 – 0
E-Mail: busik@wlp-systems.de
Internet: www.wlp-systems.de

Quellcode [1] - Objekttypen

```
create type t_ldap_person as object (
                                    -- inetOrgPerson Attribute
    per_abteilung       varchar2(32),   -- departmentNumber
    per_display_name    varchar2(1024), -- displayName
    per_mitarbeiter_nr  varchar2(32),   -- employeeNumber
    per_vorname         varchar2(100),  -- givenName
    per_initialien      varchar2(32),   -- initials
    per_email1          varchar2(256),  -- mail
    per_email2          varchar2(256),  -- mail
                                    --  nicht implementiert:  manager  (DN, auch mehrere)
    per_mobil_nr1       varchar2(32),   -- mobile
    per_mobil_nr2       varchar2(32),   -- mobile
    per_uid             varchar2(256),  -- uid
                                    -- attr. von organizationalPerson
    per_titel           varchar2(100),  -- title
    per_tel_nr1         varchar2(32),   -- telephoneNumber
    per_tel_nr2         varchar2(32),   -- telephoneNumber
    per_fax_nr          varchar2(32),   -- facsimileTelephoneNumber
                                    -- attr. von person
    per_nachname        varchar2(100),  -- sn (must)
    per_login           varchar2(100),  -- cn (must)
    per_crea_date       date            -- createTimestamp (operational Attribute)
)
create type t_ldap_person_table as table of t_ldap_person
```

Quellcode [2] – package header

```
create or replace package ldap_person as
    -- $Id: $
    -- (c) 2007 Martin Busik, WLP Systems GmbH
    --
    -- Verbindungsdaten
    --
    CONST_LDAP_HOST     constant varchar2(100) := 'localhost';
    CONST_LDAP_PORT     constant varchar2(100) := '389';
    CONST_LDAP_USER     constant varchar2(256) := 'cn=root,dc=busik,dc=de';
    CONST_LDAP_PASSWD   constant varchar2(256) := 'secret';
    --
    -- Default-Selektionskriterien
    --
    CONST_LDAP_DEFAULT_BASE constant varchar2(4096) :=
        'ou=wlps,dc=busik,dc=de';
    CONST_LDAP_DEFAULT_FILTER constant varchar2(4096) :=
        '(objectclass=inetOrgPerson)';
    CONST_LDAP_DEFAULT_SCOPE constant number := DBMS_LDAP.SCOPE_ONELEVEL;
```

Quellcode [3] – package header

```
    --
    -- Attributnamen
    --
    ATTR_DEPARTMENTNUMBER          constant varchar2(32) := 'departmentNumber';
    ATTR_DISPLAYNAME               constant varchar2(32) := 'displayName';
    ATTR_EMPLOYEENUMBER            constant varchar2(32) := 'employeeNumber';
    ATTR_GIVENNAME                 constant varchar2(32) := 'givenName';
    ATTR_INITIALS                  constant varchar2(32) := 'initials';
    ATTR_MAIL                      constant varchar2(32) := 'mail';
    ATTR_MOBILE                    constant varchar2(32) := 'mobile';
    ATTR_UID                       constant varchar2(32) := 'uid';
    ATTR_TITLE                     constant varchar2(32) := 'title';
    ATTR_TELEPHONENUMBER           constant varchar2(32) := 'telephoneNumber';
    ATTR_FACSIMILETELEPHONENUMBER  constant varchar2(32) := 'facsimileTelephoneNumber';
    ATTR_SN                        constant varchar2(32) := 'sn';
    ATTR_CN                        constant varchar2(32) := 'cn';
    ATTR_CREATETIMESTAMP           constant varchar2(32) := 'createTimestamp';
    --
    --
    -- Die Hauptfunktion - das eigentliche holen der Daten.
    --
    function    do_select (
        in_filter varchar2 default CONST_LDAP_DEFAULT_FILTER,
        in_base   varchar2 default CONST_LDAP_DEFAULT_BASE
    )
    return     t_ldap_person_table
    pipelined;
    --
END;
/
```

Quellcode [4] package body

```
create or replace package body ldap_person as
--
--  $Id: $
--  (c) 2007 Martin Busik, WLP Systems GmbH
--
--
function new_person
return t_ldap_person
is
begin
    return t_ldap_person(
        null,null,null,null,null,
        null,null,null,null,null,
        null,null,null,null,null,
        null,null );
end;
```

Quellcode [5] – package body

```
--
-- liefert die Attribute, deren Werte der LDAP-Server
-- zurückliefern soll (a la select-liste in SQL)
--
function get_attributes
return DBMS_LDAP.string_collection
is
  loc_attrs        DBMS_LDAP.string_collection;
begin
    loc_attrs(1)    := ATTR_DEPARTMENTNUMBER;
    loc_attrs(2)    := ATTR_DISPLAYNAME;
    loc_attrs(3)    := ATTR_EMPLOYEENUMBER;
    loc_attrs(4)    := ATTR_GIVENNAME;
    loc_attrs(5)    := ATTR_INITIALS;
    loc_attrs(6)    := ATTR_MAIL;
    loc_attrs(7)    := ATTR_MOBILE;
    loc_attrs(8)    := ATTR_UID;
    loc_attrs(9)    := ATTR_TITLE;
    loc_attrs(10)   := ATTR_TELEPHONENUMBER;
    loc_attrs(11)   := ATTR_FACSIMILETELEPHONENUMBER;
    loc_attrs(12)   := ATTR_SN;
    loc_attrs(13)   := ATTR_CN;
    loc_attrs(14)   := ATTR_CREATETIMESTAMP;
    return loc_attrs;
end; -- get_attributes;
```

Quellcode [6] – package body

```
--
-- Diese Prozedur enthält das Mapping der LDAP-Attribute auf die Felder
-- im T_LDAP_PERSON Typ.
-- Ebenfalls hier erfolgt die Behandlung von multivalue-Attributen
--
procedure person_setters (
    in_session  DBMS_LDAP.session,
    out_person  in out t_ldap_person,
    in_entry    DBMS_LDAP.message
) is
    loc_attrname    varchar2(256);
    loc_ber_element DBMS_LDAP.ber_element;
    --
    -- lokale Funktion/getter. Liefert eine Collection
    -- aller Werte zum geg. Attribut
    --
    function get_values(in_attrname varchar2)
    return DBMS_LDAP.string_collection
    is
        loc_result DBMS_LDAP.string_collection;
    begin
        loc_result := DBMS_LDAP.get_values (
                        ld       => in_session,
                        ldapentry => in_entry,
                        attr     => in_attrname);
        return loc_result;
    end;
```

Quellcode [7] – package body

```
--
-- lokale Funktion/getter. Liefert den ersten
-- (falls Attribut multivalue) oder einzigen
-- Inhalt des Attributes
--
function get_first_value(in_attrname varchar2)
return varchar2
is
    loc_result  varchar2(4096) := null;
    loc_vals    DBMS_LDAP.string_collection;
begin
    loc_vals := DBMS_LDAP.get_values (
                    ld        => in_session,
                    ldapentry => in_entry,
                    attr      => in_attrname);
    if loc_vals is not null and loc_vals.count > 0
    then
        loc_result := loc_vals(loc_vals.first);
    end if;
    return loc_result;
end;
```

Quellcode [8] – package body

```
-- lokale Prozedur/setter. Setzt das entsprechende Feld
-- im t_ldap_person. Wird für Attribute bzw. Felder
-- verwendet, bei denen ein 1:1 Mapping erfolgt
--
procedure scalar_set(in_attrname varchar2, in_value varchar2)
is
begin
    case in_attrname
        when ATTR_DEPARTMENTNUMBER then out_person.per_abteilung := in_value;
        when ATTR_DISPLAYNAME then out_person.per_display_name := in_value;
        when ATTR_EMPLOYEENUMBER then out_person.per_mitarbeiter_nr := in_value;
        when ATTR_GIVENNAME then out_person.per_vorname := in_value;
        when ATTR_INITIALS then out_person.per_initialien := in_value;
        when ATTR_UID then out_person.per_uid := in_value;
        when ATTR_TITLE then out_person.per_titel := in_value;
        when ATTR_FACSIMILETELEPHONENUMBER then out_person.per_fax_nr := in_value;
        when ATTR_SN then out_person.per_nachname := in_value;
        when ATTR_CN then out_person.per_login := in_value;
        else
            dbms_output.put_line('UNKNOWN MAPPING FOR: '||in_attrname);
    end case;
end;
```

Quellcode [9] – package body

```
--
-- lokale Prozedur/setter. Setzt mehrere Felder
-- von t_ldap_person durch Werte aus einer Multivalue-Collection
--
procedure multifield_setter (
    in_values DBMS_LDAP.string_collection,
    out_field1 out varchar2,
    out_field2 out varchar2
) is
begin
    if in_values is not null and in_values.count > 0
    then
        out_field1 := in_values(in_values.first);
        --
        if in_values.count > 1
        then
            out_field2 := in_values(in_values.next(in_values.first));
        end if;
    end if;
end;
--
```

Quellcode [10] – package body

```
begin
    loc_attrname := DBMS_LDAP.first_attribute(
                        ld        => in_session,
                        ldapentry => in_entry,
                        ber_elem  => loc_ber_element);
    --
    while loc_attrname IS NOT NULL
    loop
        case loc_attrname
            when ATTR_MAIL then
                    multifield_setter(get_values(loc_attrname),
                        out_person.per_email1,out_person.per_email2);
            when ATTR_MOBILE then
                    multifield_setter(get_values(loc_attrname),
                        out_person.per_mobil_nr1, out_person.per_mobil_nr2);
            when ATTR_TELEPHONENUMBER then
                    multifield_setter(get_values(loc_attrname),
                        out_person.per_tel_nr1, out_person.per_tel_nr2);
            when ATTR_CREATETIMESTAMP then
                out_person.per_crea_date :=
                    to_date(substr(get_first_value(loc_attrname),1,14),'YYYYMMDDHH24MISS');
            else
                scalar_set(loc_attrname,get_first_value(loc_attrname));
        end case;
        --
```

Quellcode [11] – package body

```
 --
        -- Iteration über alle (gelieferten) Attribute
        --
        loc_attrname := DBMS_LDAP.next_attribute(
                             ld        => in_session,
                             ldapentry => in_entry,
                             ber_elem  => loc_ber_element);
    end loop;
end;
```

Quellcode [12] – package body

```
--
-- Hauptfunktion
--   in_filter - ein gültiges Filterkriterium. Default siehe package header
--   in_base   - Startpunkt für die Suche, Default siehe package header
--
-- Rückgabewert:
--   liefert eine Tabelle von t_ldap_person Säzen als Ergebnis der
--   LDAP-Suche
--
function    do_select (
    in_filter varchar2 default CONST_LDAP_DEFAULT_FILTER,
    in_base   varchar2 default CONST_LDAP_DEFAULT_BASE
)
return     t_ldap_person_table
pipelined
is
    loc_person      t_ldap_person;
    loc_retval      PLS_INTEGER;
    loc_session     DBMS_LDAP.session;
    loc_message     DBMS_LDAP.message;
    loc_entry       DBMS_LDAP.message;
begin
    --
    -- im Fehlerfall soll das LDAP-Package eine Exception werfen
    --
    DBMS_LDAP.USE_EXCEPTION := TRUE;
```

Quellcode [13] – package body

```
--
-- Connect
--
loc_session := DBMS_LDAP.init(hostname => CONST_LDAP_HOST,
                  portnum  => CONST_LDAP_PORT);
loc_retval  := DBMS_LDAP.simple_bind_s(ld     => loc_session,
                  dn     => CONST_LDAP_USER,
                  passwd => CONST_LDAP_PASSWD);
--
-- LDAP unterstützt auch eine anonyme Anmeldung, diese würde
-- wie folgt aufgebaut werden:
--
-- loc_retval  := DBMS_LDAP.simple_bind_s(ld => loc_session,
--                  dn => null,passwd => null);
--
-- Suche
--
loc_retval := DBMS_LDAP.search_s(
                ld       => loc_session,
                base     => in_base,
                scope    => CONST_LDAP_DEFAULT_SCOPE,
                filter   => in_filter,
                attrs    => get_attributes,
                attronly => 0,
                res      => loc_message);
```

Quellcode [14] – package body

```
if loc_retval = DBMS_LDAP.SUCCESS and loc_message is not null
    then
        loc_entry := DBMS_LDAP.first_entry(ld  => loc_session,
                                           msg => loc_message);
        while loc_entry IS NOT NULL
        loop
            loc_person := new_person;
            person_setters(loc_session,loc_person,loc_entry);
            --
            -- Einen Satz (=t_ldap_person) zurückgeben
            --
            pipe row(loc_person);
            --
            -- Iteration über alle Treffer
            --
            loc_entry := DBMS_LDAP.next_entry(ld  => loc_session,
                                              msg => loc_entry);
        end loop;
    end if;
    --
    -- disconnect
    --
    loc_retval := DBMS_LDAP.unbind_s(ld => loc_session);
    return;
end; -- function select
```

Quellcode [15] – package body

```
--
end; -- package body
/

show errors package body ldap_person

-- debugging/Verwendung
-- set pagesize 100
-- select * from table(ldap_person.do_select);
-- select per_nachname,per_email1,per_email2,per_crea_date,per_login from table(ldap_person.do_select);
select per_nachname,per_email1,per_email2,per_crea_date,per_login from table(ldap_person.do_select('mail=*busik.de'));
```