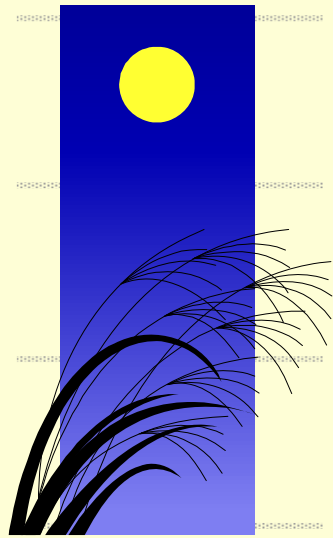
スキーマってなんだろう

2007/10/06 OSC2007Tokyo/Fall

日本LDAPユーザー会

太田俊哉



今日の内容

- LDAPのスキーマについて説明します
- LDAPのデータ構造などについて説明します
- やや観念的な話が中心です

混沌から整理へ

江津さん

123-4567

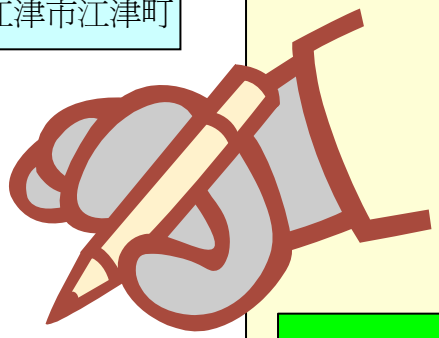
109-3434.

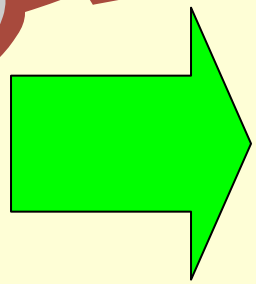
goutu@sancou....

島根県江津市江津町

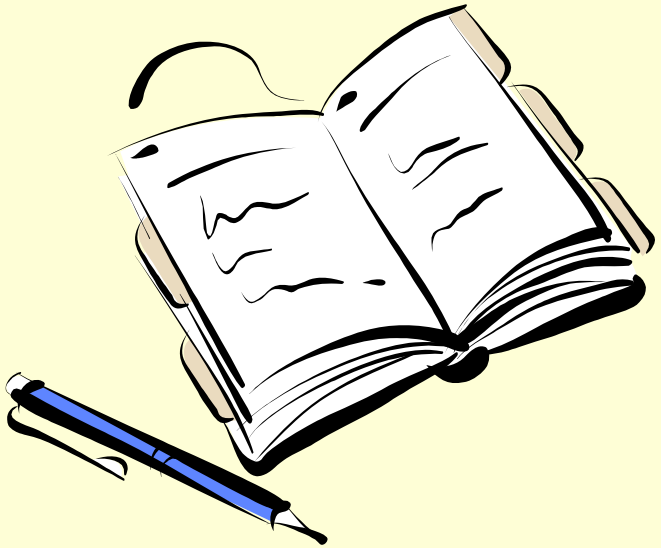
goutu-01@gmail..

enoca@anet.n..

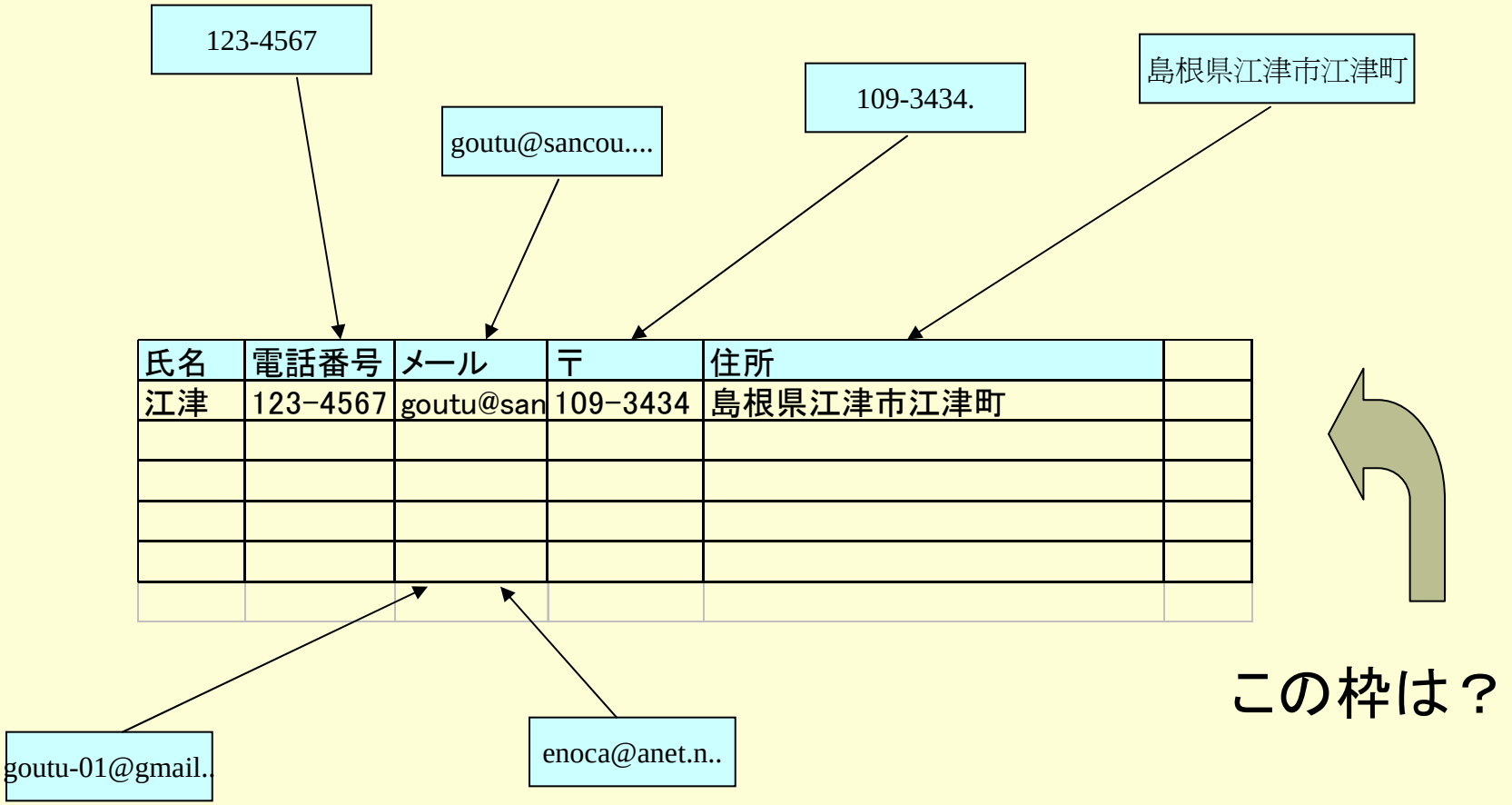




電話番号	〒	住所	メール



枠の中に入れる


123-4567
goutu@sancou....
109-3434.
島根県江津市江津町
氏名
電話番号
メール
〒
住所
江津
123-4567
goutu@san
109-3434
島根県江津市江津町
goutu-01@gmail..
enoca@anet.n..
この枠は？

枠の意味

- 1つのデータにはいろいろな項目がある
- ある目的(たとえば住所録)でデータを集めるときには項目はおのずと決まる
- データの構造を決める
  ……スキーマ

器と構造

LDAPのスキーマ

- いろいろな用途に合わせ、標準でいろいろなものが提供されている
- 特別な応用例では、別途提供されている場合もある(例: Sambaとの連携)
- もちろん自分で作ることも可能
(若干ルールはある)

LDAPのスキーマの例

- 電話帳が一番わかりやすい
  - 個人に関わる情報(氏名、住所、電話番号、メールアドレスなどなど)
  - 個人⇒主体、個人にかかわる情報⇒属性
  - 情報は属性とデータの組
  - 属性は複数個存在、1つの属性に複数の値もありえる(ここが大事)

標準スキーマ

- あらかじめLDAPサーバに準備されているもの
- OpenLDAPだと以下の通り

ファイル	説明
core.schema	OpenLDAP core (必須)
cosine.schema	Cosine and Internet X.500 (有用)
inetorgperson.schema	InetOrgPerson (有用)
misc.schema	Assorted (実験用)
nis.schema	Network Information Services (参考)
openldap.schema	OpenLDAP Project (実験用)

枠....ではない

- データを枠の中に入れる
  - 考え方は合っているが、LDAPの枠は表ではない
  - LDAP≠表形式データベース
  - データベースでよくやる正規化はしません
- データベースの一種ではあるが
  - 木構造
  - データは手繰っていく

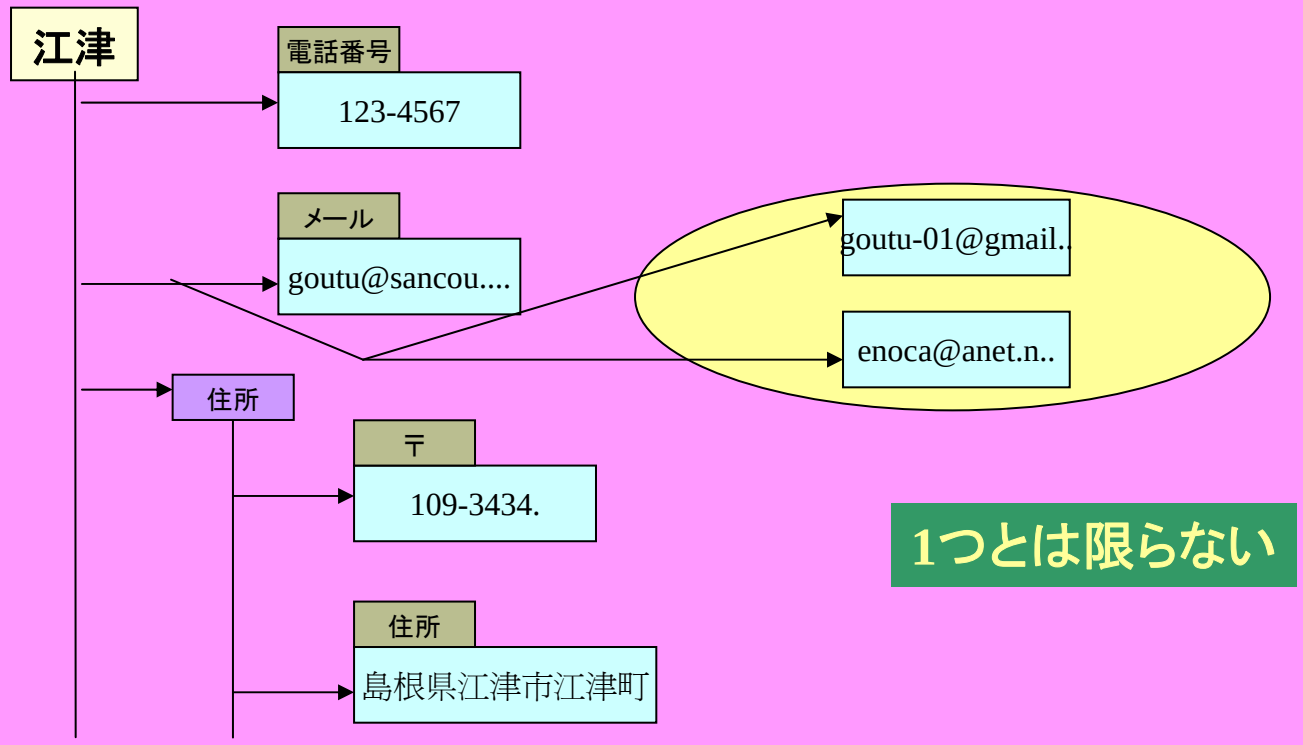
図にしてみると

氏名	電話番号	メール	〒	住所
江津	123-4567	goutu@san	109-3434	島根県江津市江津町
川平	123-4579	kawahira@s	109-3434	島根県江津市川平町
川戸	123-5432	kawato@sa	109-3434	島根県江津市桜江町
鹿賀	133-1222	shikaga@sa	109-3434	島根県江津市桜江町
明塚	142-1928	akatsuka@s	109-3434	島根県邑智郡美郷町
浜原	118-8791	hamahara@	109-3434	島根県邑智郡美郷町
沢谷	109-0122	sawatani@s	109-3434	島根県邑智郡美郷町

普通はこのような表形式

データベースとしては一般的

LDAPではこちら



1つとは限らない

データベース=表という固定概念

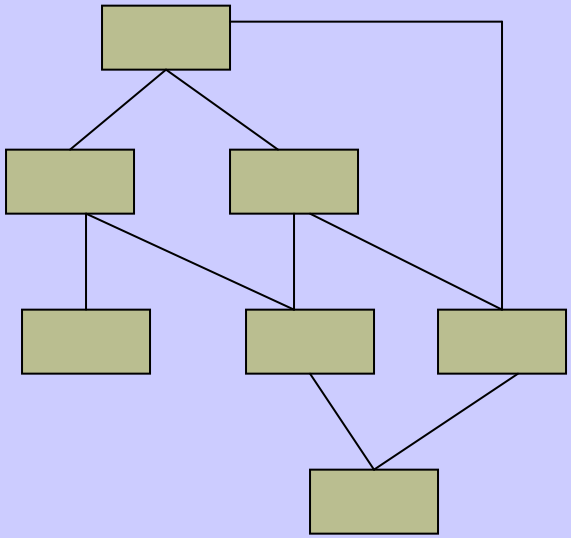
リレーショナルデータベース

seq	id	pass	user
100	hoge	***	hoge
101	foo	***	test
102	bar	***	gues

seq	home
100	/home/hoge
101	/home2/var

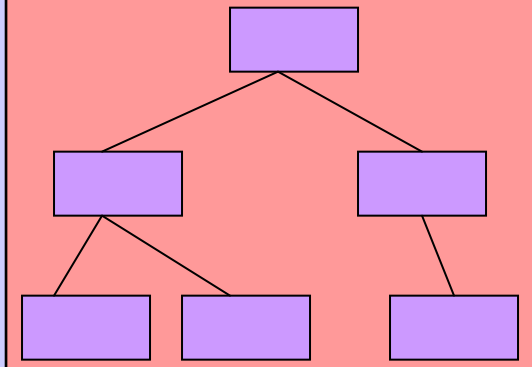
●表中にデータ、表間での関係、表への演算
●一般的

CDASYL型データベース (ネットワーク型データベース)



●ノードが繋がっている
●親へのリンク

階層型データベース



●木構造
●アクセスパスは1つ
●LDAPはこれ

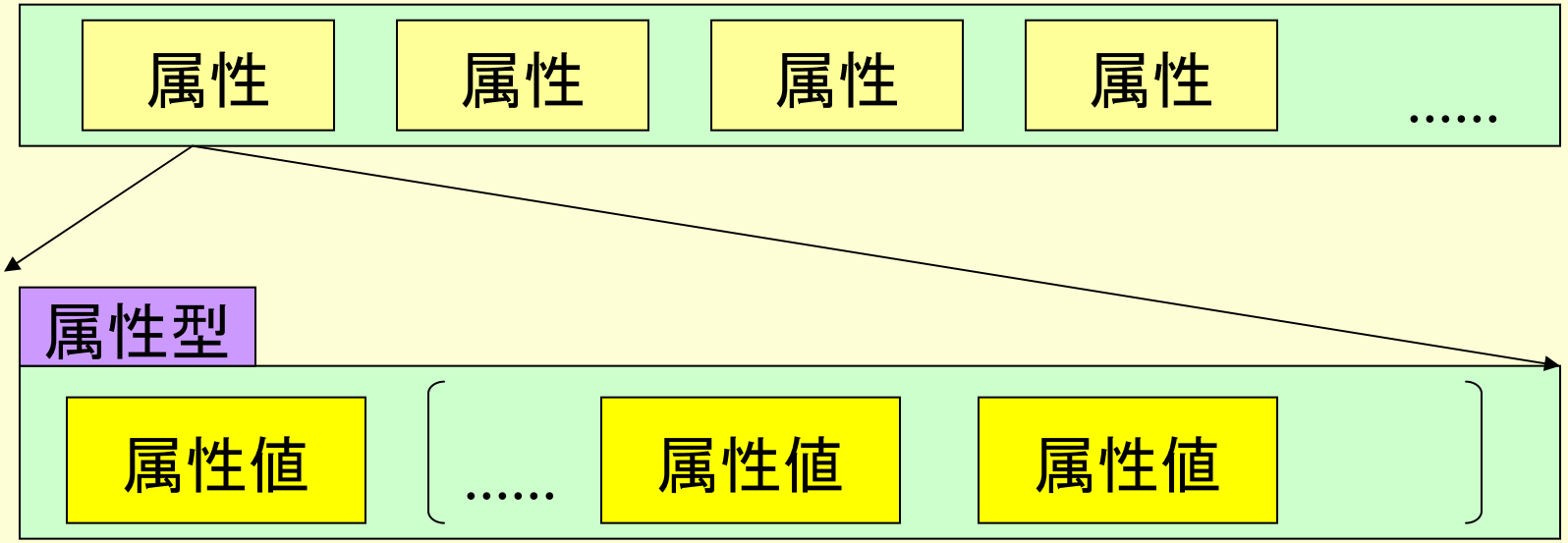
何がほしい?

- 情報だ
- LDAPは検索が中心
  - 更新頻度は少ない
  - 早く検索できることが大事⇒情報が早く欲しい
  - 更新直後に新しいデータが検索できる必要もない
- ではLDAPのデータ構造は?
  - スキーマの中身と言い換えてもよいかも

では、中身を見ていきましょう

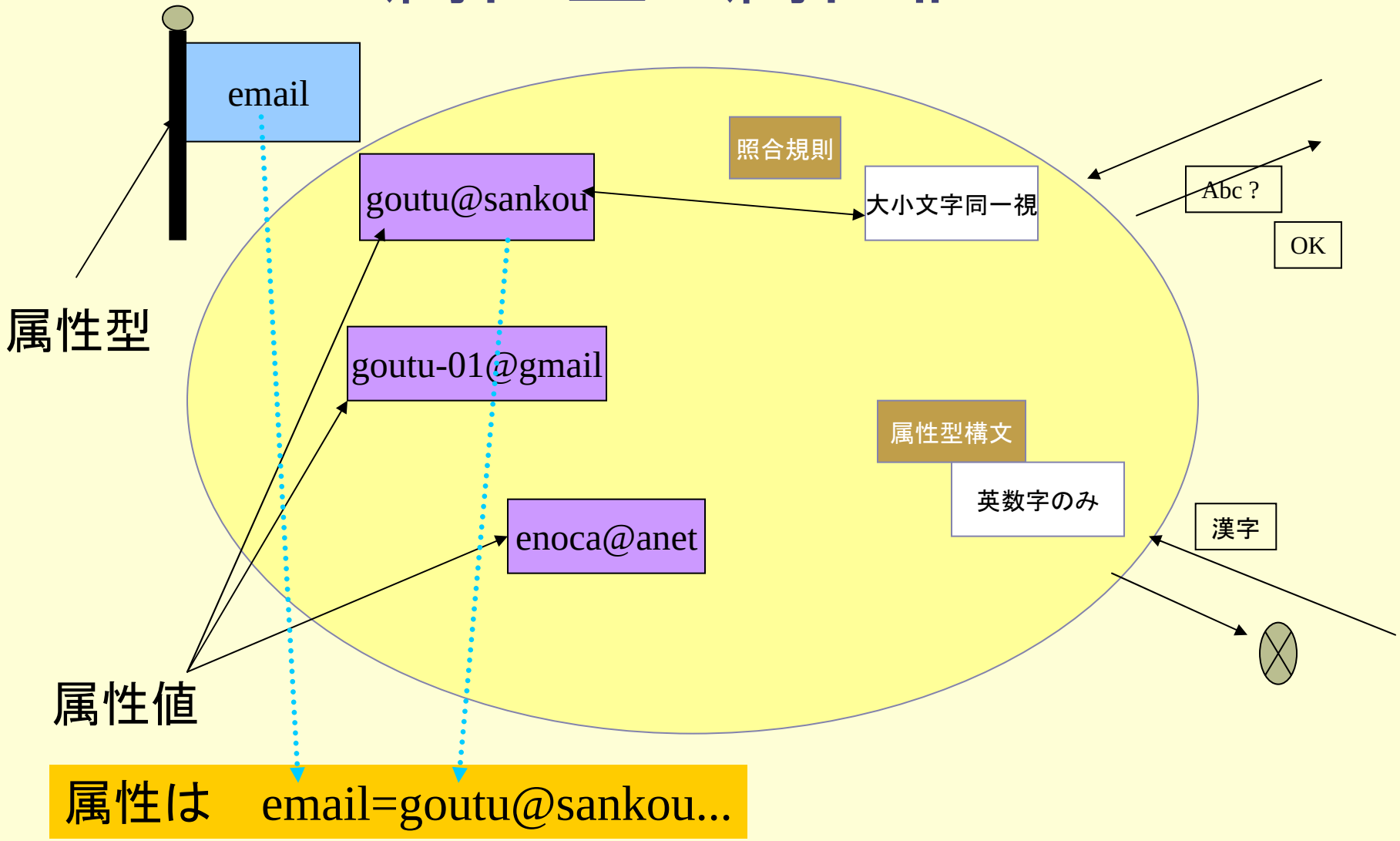
- いくつかのキーワード
  - エントリ
  - 属性
  - 属性値
  - ディレクトリ属性ツリー（DIT）

エントリ



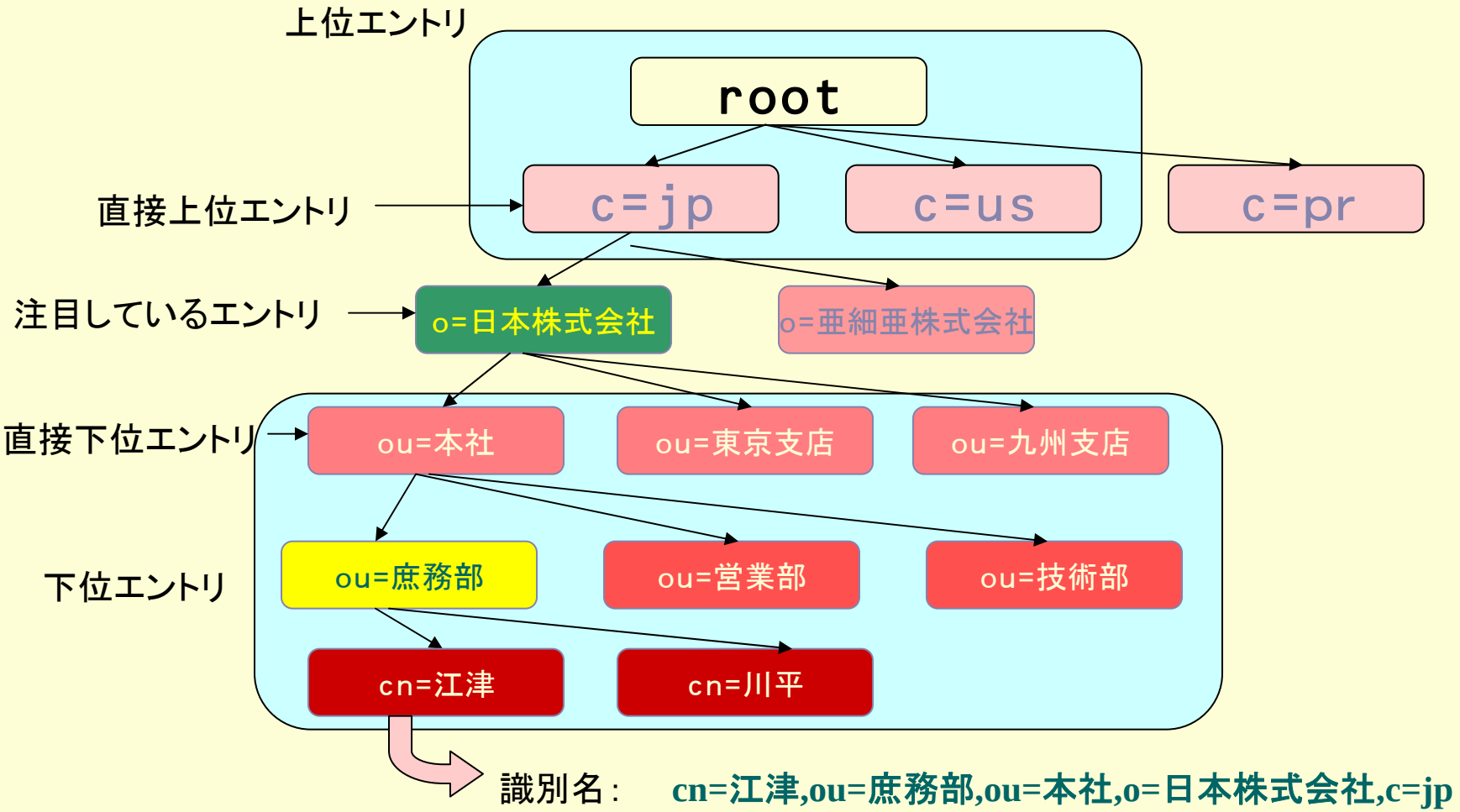
1つのエントリ⇒実世界でのあるもの（オブジェクト）
データベースの表だったらある1行
エントリには複数の属性がある
　住所、電話番号、メールアドレスなど
属性＝属性型と属性値で構成
属性型＝属性の名前
1つの属性には**複数の属性値があってもよい**

属性型と属性値



属性は　email=goutu@sankou...

DIT(Directory Information Tree)



DIT:エントリ間の階層構造

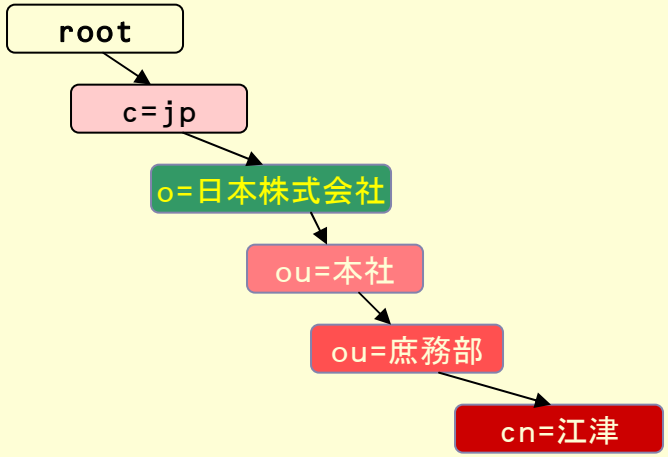
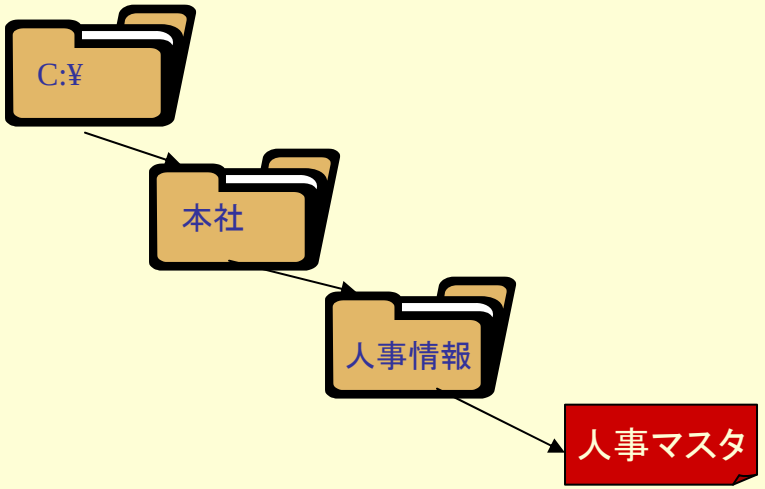
識別名と相対識別名

- 相対識別名(RDN: Relative Distinguished Name)
  - エントリの中で、一意になるようにつけられる名前
    - 例：cn=江津
- 識別名(DN: Distinguished Name)
  - あるディレクトリ情報ツリーの中で、そのエントリを一意に表わすための名前
  - あるエントリのRDNを、ルートエントリまでの間にあるエントリのRDNを下位から上位に向かって連結したもの
    - 例：cn=江津,ou=庶務部,ou=本社,o=日本株式会社,c=jp

エントリと相対識別名

- 相対識別名
  - ou=庶務部
- エントリ
  - 属性の集合⇒属性:属性値の集合
    - 属性:ou
    - 属性値:庶務部
    - 相対識別名と同じ
  - エントリの名前(識別名)は、ある属性:属性値と同じ

比べてみよう

LDAPのDIT	ファイルシステムのディレクトリ
	
・各エントリ毎に属性型と属性値が存在 ・エントリそのものが値を持つ ・エントリ≠器 ・ある特定の情報⇒識別名 ・特定のエントリ⇒相対識別名	・ディレクトリそのものは値を持たない ・ディレクトリは器 ・ある特定の情報⇒フルパス

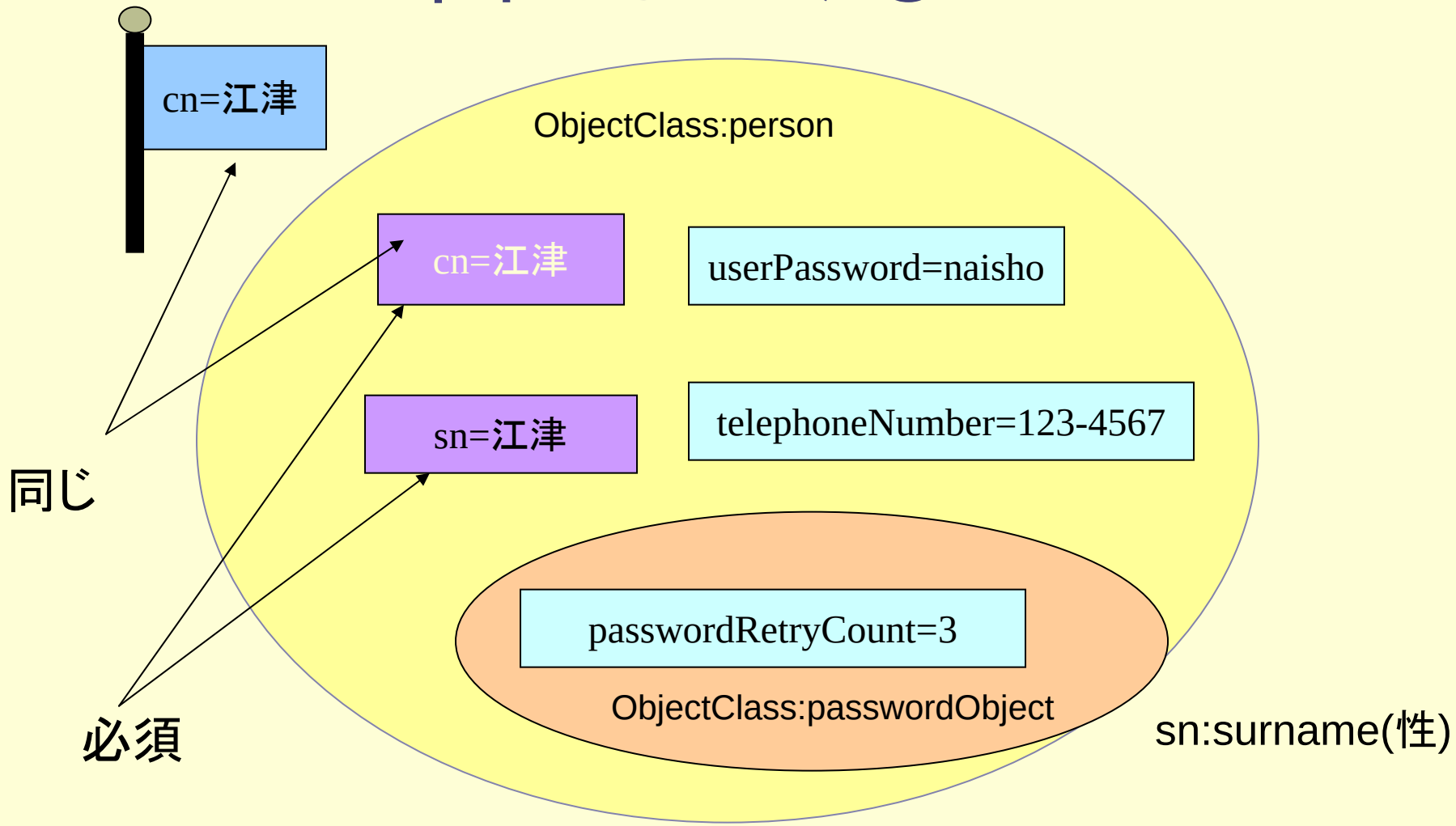
エントリの属性は任意?

- 一定のルールがあります
- オブジェクトクラス
  - エントリの構造を規定
    - 構造型オブジェクトクラス(structual objectClass)
      - MUST属性
      - MAY属性
    - 補助オブジェクトクラス(auxiliary objectClass)
    - topは例外
  - その正体は OID(Object ID)

たとえば人を表わす時には

- オブジェクトクラス person
  - MUST属性
    - cn、sn
  - MAY属性
    - userPassword、telephoneNumber、seeAlso、description
  - 補助オブジェクトクラス
    - passwordObject
      - passwordExpirationTime、passwordExpWarned、passwordRetryCount、retryCountResetTime など
- 階層構造を取る
  - 上から下へ、属性が引き継がれる

図にしてみると


cn=江津
ObjectClass:person
cn=江津
userPassword=naisho
sn=江津
telephoneNumber=123-4567
同じ
passwordRetryCount=3
ObjectClass:passwordObject
必須
sn:surname(性)

照合規則

- 属性の比較照合の時に、どのような照合条件を満たすとするかを規定
  - 同値性 (EQUALITY)
    - caseExactMatch、caseIgnoreMatchなど
  - 順序性 (ORDERING)
    - caseExactOrderingMatch、caseIgnoreOrderingMatchなど
  - 部分文字列性 (SUBSTRING)
    - caseExactSubstringsMatch、caseIgnoreSubstringsMatchなど
- 検索時に使います

属性構文

- ある属性値が持つことのできる値の定義
- その定義に合っていないと値が入らない
- 入力値チェックみたいなもの
- 例
  - INTEGER
  - Numeric String
  - など

標準定義での例

- 属性型name と cn(OpenLDAPのドキュメントより)

```
attributeType ( 2.5.4.41 NAME 'name'
        DESC 'name(s) associated with the object'
        EQUALITY caseIgnoreMatch
        SUBSTR caseIgnoreSubstringsMatch
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.15{32768} )
 attributeType ( 2.5.4.3 NAME ( 'cn' 'commonName' )
        DESC 'common name(s) asscіated with the object'
        SUP name )
```

定義方法の実際

- 使用目的に応じたディレクトリ構造の設計
- 標準スキーマの利用
- データの準備
- LDAP用データに変換

LDIF

- LDAP Interchange Format(RFC2849)
- LDAPの設定情報と内容を保存、交換するためのテキスト形式
- LDAPサーバ間で互換
- 一括登録などに使用

LDIFの形式

- テキストファイル
- エントリ記述の集合体
- dn: hogehoge....で始まる
- エントリ記述を列挙
- コマンドも書ける

LDIFの例

```
dn: dc=my-domain,dc=com
objectClass: dcObject
objectClass: organization
dc: my-domain
o: my-domain

dn: ou=People,dc=my-domain,dc=com
objectClass: organizationalUnit
ou: People

dn: uid=ldapuser,ou=People,dc=my-domain,dc=com
objectClass: account
objectClass: posixAccount
uid: ldapuser
cn: ldapuser
userPassword: ldapuser
loginShell: /bin/bash
uidNumber: 1000
gidNumber: 1000
homeDirectory: /home/ldapuser
```

直接スキーマを操作する場合

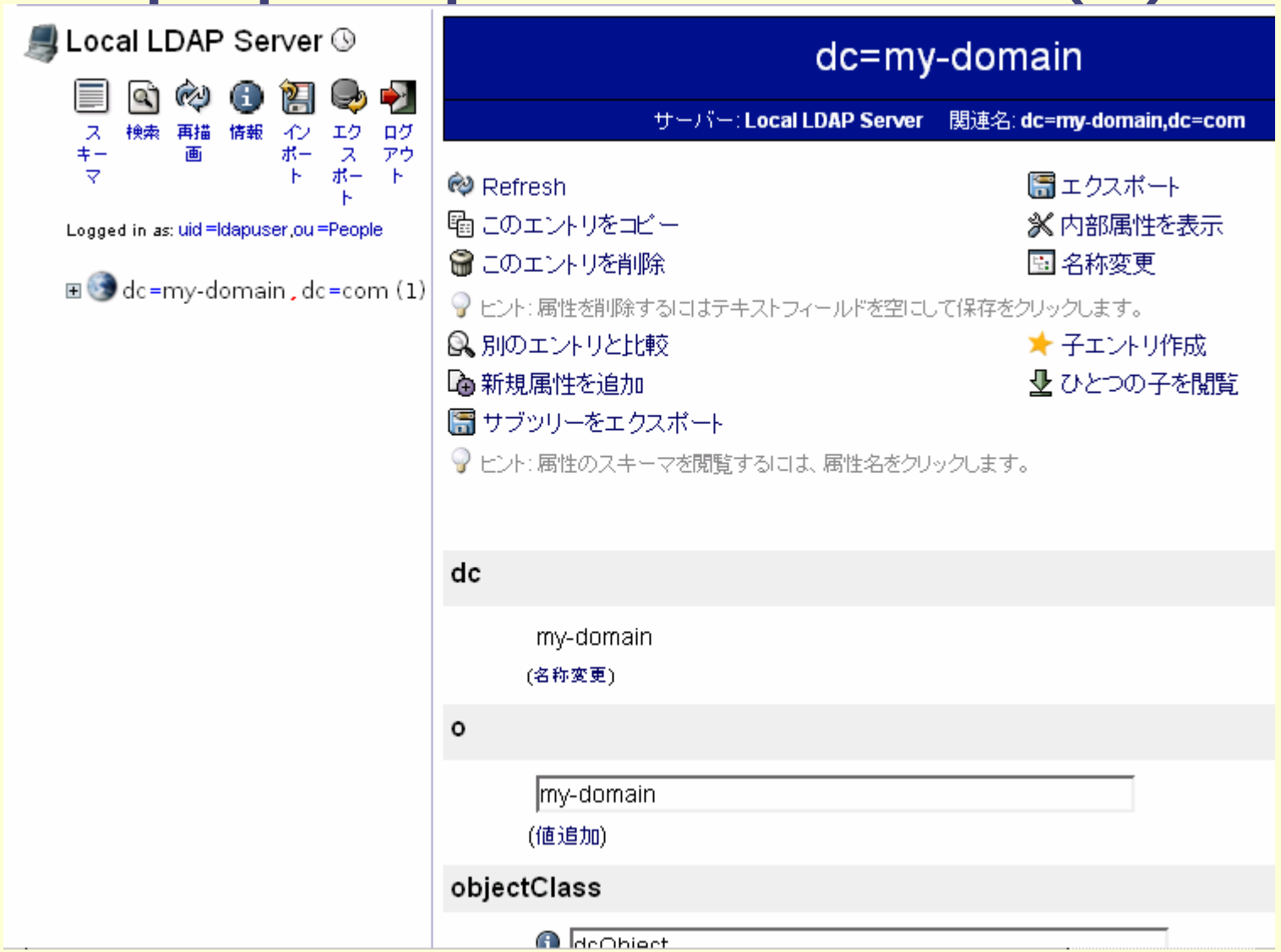
- 基本的にはありえない
- 手で、動的に修正すると、痕跡が残らない
- CSVなどで元データを作り登録するのが基本
  - ⇒LDIFファイル
- 状況を見ることはありえる

閲覧ツール

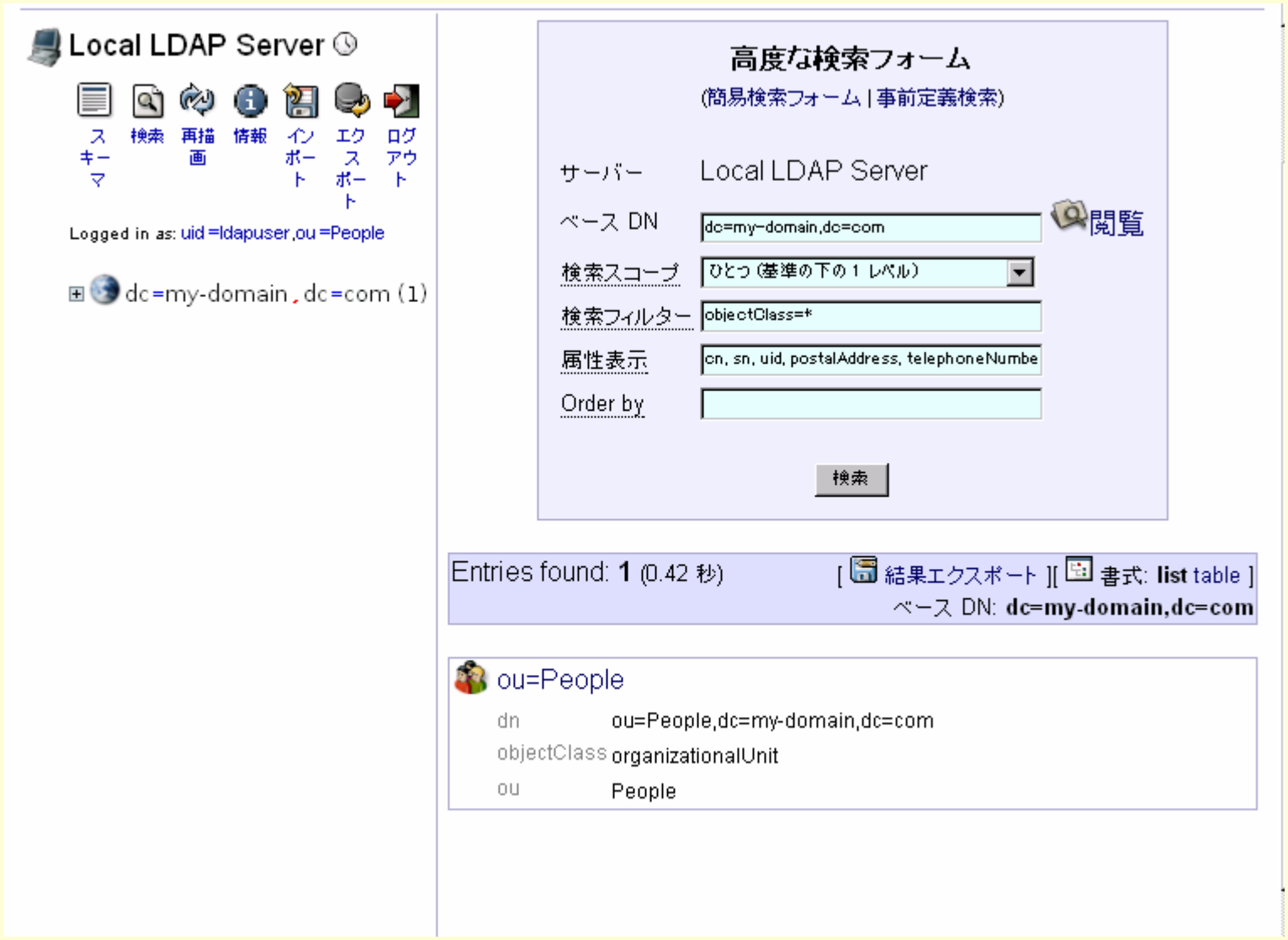
- 商用LDAP製品にはたいていある
  - 差別化のためについている
- フリーのもの
  - OpenLDAPにはない(!)
  - フリーでいくつかある
- 特定の用途に特化したもの
  - LAM(Ldap Account Manager)

phpldapadminでの例(1)

phpldapadminでの例(2)



phpldapadminの例(3)



LDAPのスキーマとは

- 木構造ではあるが、ファイルシステムとは違う
- DNとかRDNとかの独特な概念
- 属性、それに附属する属性構文など規則が多数
- 定義するのは大変
- あらかじめできているのをうまく利用

情報を共有しましょう

- LDAPの情報はそれほど多くない
  - Sambaよりは圧倒的に少ない
  - 個人で使うものではないので
  - LDAPユーザー会
  - MLで活動中です
  - 日経Linuxにも連載中
  - 是非ご参加を

Fin