Red Hat Directory Server 12

インデックスの管理

インデックスの最適化による検索パフォーマンスの向上

Last Updated: 2024-09-19

Red Hat Directory Server 12 インデックスの管理

インデックスの最適化による検索パフォーマンスの向上

法律上の通知

概要

インデックス付けは、属性または値を分類および整理することにより、情報の検索と取得を容易にします。仮想リストビューコントロールを使用することで、大規模な検索結果の連続したサブセットを要求できます。

目次

RED HAT DIRECTORY SERVER に関するフィードバックの提供

Red Hat のドキュメントおよび製品に関するご意見をお待ちしております。ドキュメントの改善点があればお知らせください。以下の方法で送信してください。

- Jira を通じて Red Hat Directory Server ドキュメントに関するフィードバックを送信する場合 (アカウントが必要):
  1. Red Hat Issue Tracker にアクセスしてください。
  2. **Summary** フィールドにわかりやすいタイトルを入力します。
  3. **Description** フィールドに、ドキュメントの改善に関するご意見を記入してください。ドキュメントの該当部分へのリンクも追加してください。
  4. ダイアログの下部にある **Create** をクリックします。
- Jira を通じて Red Hat Directory Server 製品に関するフィードバックを送信する場合 (アカウントが必要):
  1. Red Hat Issue Tracker にアクセスしてください。
  2. **Create Issue** ページで、**Next** をクリックします。
  3. **Summary** フィールドに入力します。
  4. **Component** フィールドでコンポーネントを選択します。
  5. **Description** フィールドに以下の内容を入力します。
     a. 選択したコンポーネントのバージョン番号。
     b. 問題を再現するための手順、または改善のための提案。
  6. **Create** をクリックします。

第1章 新規に作成されたすべてのデータベースに適用されるデフォルトインデックスの定義

Directory Server のデフォルトのインデックスでは、インデックスを作成する属性のセットが定義されています。新しいデータベースを作成すると、Directory Server はデフォルトのインデックス属性を **cn=default indexes,cn=config,cn=ldbm database,cn=plugins,cn=config** エントリーからデータベース固有の **cn=index,cn=database_name,cn=ldbm database,cn=plugins,cn=config** エントリーにコピーします。

注記

Directory Server は、デフォルトのインデックスの変更を既存のデータベースに適用しません。

1.1. インデックスの種類

Directory Server は、インデックス付きの各属性のインデックスを、インスタンスのデータベースディレクトリー内の個別のデータベースファイルに保存します。たとえば、**sn** 属性のインデックスは、**/var/lib/dirsrv/slapd-instance_name/db/database_name/sn.db** ファイルに格納されています。Directory Server が 1 つの属性に対して異なるインデックスを維持している場合、各インデックスファイルには複数のインデックスタイプを含めることができます。

Directory Server は以下のインデックスタイプをサポートしています。

- 存在インデックス (**pres**) は、特定の属性を含むエントリーのリストです。たとえば、**attribute=mail** のようにクライアントが頻繁に検索を行う場合はこのタイプを使用します。
- 等価インデックス (**eq**) により、特定の属性値を含むエントリーの検索が改善されます。たとえば、**cn** 属性に等価インデックスを設定すると、**cn=first_name last_name** の検索を高速に行うことができます。
- 近似インデックス (**approx**) は、効率的な近似検索や音符のような検索を可能にします。たとえば、**cn~=first_name last_name**、**cn~=first_name**、または **cn~=first_nam** (スペルに注意) は、**cn=first_name X last_name** というエントリーが返されます。Directory Server の metaphone 表音アルゴリズムは US-ASCII 文字のみをサポートします。したがって、近似インデックスは、英語の値でのみ使用してください。
- 部分文字列インデックス (**sub**) は、維持するためのコストがかかるインデックスですが、エントリー内の部分文字列に対して効率的な検索が可能になります。部分文字列のインデックスは、各エントリーの最小 3 文字に制限されます。たとえば、**telephoneNumber=*555*** と検索すると、**telephoneNumber** 属性に **555** を含む値を持つディレクトリー内のすべてのエントリーが返されます。
- 国際インデックスは、国際ディレクトリー内の情報の検索を迅速化します。国際インデックスの作成プロセスは、通常のインデックスを作成するプロセスと似ています。ただし、オブジェクト識別子 (OID) をインデックス化する属性に関連付けることで一致するルールを適用する点が異なります。

1.2. インデックスのメリットとのバランス

新しいインデックスを作成する前に、インデックスを維持することのメリットとコストのバランスを考えてみます。

- 近似インデックスは、通常、数字を含む属性 (電話番号など) には効率的ではありません。
- 部分文字列のインデックスはバイナリー属性では機能しません。
- イメージなど、大きな値を含む属性の等価インデックスは回避してください。
- 検索で一般的に使用されない属性のインデックスを維持すると、検索パフォーマンスを向上させることなくオーバーヘッドが増加します。
- インデックス化されていない属性は、検索要求で依然として使用できますが、検索のタイプによっては検索パフォーマンスが大幅に低下する可能性があります。

インデックスは、非常に時間がかかります。たとえば、Directory Server が追加操作を受け取った場合、Directory Server はインデックス属性を調べて、属性値に対してインデックスが維持されているかどうかを判断します。作成された属性値がインデックス化されている場合、Directory Server は新しい属性値をインデックスに追加し、その後、実際の属性値がエントリーに作成されます。

例1.1 ユーザーがエントリーを追加する際に Directory Server が行うインデックス作成手順

Directory Server が以下のインデックスを維持していると仮定します。

- **cn** および **sn** 属性の等値性、近似性、および部分文字列インデックス。
- **telephoneNumber** 属性の等価および部分文字列のインデックス。
- **description** 属性の部分文字列インデックス。

たとえば、ユーザーが次のようなエントリーを追加したとします。

```
dn: cn=John Doe,ou=People,dc=example,dc=com
objectclass: top
objectClass: person
objectClass: orgperson
objectClass: inetorgperson
cn: John Doe
cn: John
sn: Doe
ou: Manufacturing
ou: people
telephoneNumber: 408 555 8834
description: Manufacturing lead
```

ユーザーがエントリーを追加すると、Directory Server は以下のステップを実行します。

1. **John** および **John Doe** の **cn** 等価インデックスエントリーを作成します。
2. **John** および **John Doe** の適切な **cn** の近似インデックスエントリーを作成します。
3. **John** および **John Doe** の **cn** 部分文字列インデックスエントリーを作成します。
4. **Doe** の **sn** 等価インデックスエントリーを作成します。
5. **Doe** に対する **sn** 近似インデックスエントリーを作成します。
6. **Doe** の **sn** 部分文字列インデックスエントリーを作成します。
7. **408 555 8834** の **telephoneNumber** 等価インデックスエントリーを作成します。

8. **408 555 8834** の **telephoneNumber** 部分文字列インデックスエントリーを作成します。

9. **Manufacturing lead** の **description** 部分文字列インデックスエントリーを作成します。

この例は、大規模なディレクトリーのデータベースを作成および保守するために必要なアクションの数が、非常に多くのリソースを消費する可能性があることを示しています。

重要

Directory Server のパフォーマンスに影響を与える可能性があるため、メンバーシップ属性 (**member**、**uniquemember** など) の部分文字列インデックスを定義しないでください。メンバーを追加または削除する場合 (たとえば、多数のメンバーを持つグループに **uniquemember** を追加する場合)、**uniquemember** の部分文字列インデックスの計算では、追加または削除された値だけでなく、すべての **uniquemember** 値を評価する必要があります。

1.3. デフォルトのインデックス属性

Directory Server は、デフォルトのインデックス属性を、**cn=default indexes,cn=config,cn=ldbm database,cn=plugins,cn=config** エントリーに格納します。インデックスタイプを含めて表示するには、次のように入力します。

```
# ldapsearch -D "cn=Directory Manager" -W -H ldap://server.example.com -b "cn=default indexes,cn=config,cn=ldbm database,cn=plugins,cn=config" -s one -o ldif-wrap=no
```

表1.1 Directory Server のデフォルトインデックス属性

aci	**cn**	**entryUSN**
entryUUID	**givenName**	**mail**
mailAlternateAddress	**mailHost**	**member**
memberOf	**nsUniqueId**	**nsCertSubjectDN**
nsTombstoneCSN	**ntUniqueId**	**ntUserDomainId**
numSubordinates	**objectClass**	**owner**
parentId	**seeAlso**	**sn**
targetUniqueId	**telephoneNumber**	**uid**
uniqueMember		

警告

表に記載されている属性 (システムインデックス) をデータベースのインデックスから削除すると、Directory Server のパフォーマンスに大きな影響を与える可能性があります。

1.4. デフォルトインデックスの維持

Directory Server は、デフォルトのインデックス属性を、**cn=default indexes,cn=config,cn=ldbm database,cn=plugins,cn=config** エントリーに格納します。なお、デフォルトのインデックス属性を維持できるのは、LDIF ステートメントを使用した場合のみです。

手順

- たとえば、インデックスタイプが **eq** と **sub** のデフォルトインデックスに **roomNumber** 属性を追加するには、次のように入力します。

```
# ldapadd -D "cn=Directory Manager" -W -H ldap://server.example.com -x

dn: cn=roomNumber,cn=default indexes,cn=config,cn=ldbm database,cn=plugins,cn=config
objectClass: nsIndex
objectClass: top
cn: roomNumber
nsSystemIndex: false
nsIndexType: eq
nsIndexType: sub
```

  以下は、LDIF ステートメントの説明です。

  - **objectClass: nsIndex**: このエントリーがインデックスエントリーであることを定義します。
  - **objectClass: top**: このオブジェクトクラスは、インデックスエントリーで追加的に必要となります。
  - **cn**: インデックスに属性の名前を設定します。
  - **nsSystemIndex**: インデックスが Directory Server の運用に必須であるかどうかを示します。
  - **nsIndexType**: この多値属性は、インデックスタイプを指定します。

- たとえば、**roomNumber** 属性のデフォルトのインデックス属性に **pres** インデックスタイプを追加するには、次のように入力します。

```
# ldapmodify -D "cn=Directory Manager" -W -H ldap://server.example.com -x

dn: cn=roomNumber,cn=default indexes,cn=config,cn=ldbm database,cn=plugins,cn=config
changetype: modify
add: nsIndexType
nsIndexType: pres
```

- たとえば、**roomNumber** 属性のデフォルトのインデックス属性から **pres** インデックスタイプを削除するには、次のように入力します。

```
# ldapmodify -D "cn=Directory Manager" -W -H ldap://server.example.com -x

dn: cn=roomNumber,cn=default indexes,cn=config,cn=ldbm database,cn=plugins,cn=config
changetype: modify
delete: nsIndexType
nsIndexType: pres
```

- たとえば、デフォルトのインデックスから **roomNumber** 属性を削除するには、次のように入力します。

```
# ldapdelete -D "cn=Directory Manager" -W -H ldap://server.example.com -x
cn=roomNumber,cn=default indexes,cn=config,cn=ldbm
database,cn=plugins,cn=config
```

検証

- デフォルトのインデックス属性をリスト表示して、変更を確認します。

```
# ldapsearch -H ldap://server.example.com:389 -D "cn=Directory Manager" -W -b
"cn=default indexes,cn=config,cn=ldbm database,cn=plugins,cn=config" -x -s one -o
ldif-wrap=no
```

第2章 特定のデータベースのインデックスを維持する

Directory Server の各データベースには、それぞれインデックスがあります。インデックスの作成、更新、削除は、**dsconf** ユーティリティーや Web コンソールを使用して行うことができます。

2.1. インデックスの種類

Directory Server は、インデックス付きの各属性のインデックスを、インスタンスのデータベースディレクトリー内の個別のデータベースファイルに保存します。たとえば、**sn** 属性のインデックスは、**/var/lib/dirsrv/slapd-instance_name/db/database_name/sn.db** ファイルに格納されています。Directory Server が 1 つの属性に対して異なるインデックスを維持している場合、各インデックスファイルには複数のインデックスタイプを含めることができます。

Directory Server は以下のインデックスタイプをサポートしています。

- 存在インデックス (**pres**) は、特定の属性を含むエントリーのリストです。たとえば、**attribute=mail** のようにクライアントが頻繁に検索を行う場合はこのタイプを使用します。
- 等価インデックス (**eq**) により、特定の属性値を含むエントリーの検索が改善されます。たとえば、**cn** 属性に等価インデックスを設定すると、**cn=first_name last_name** の検索を高速に行うことができます。
- 近似インデックス (**approx**) は、効率的な近似検索や音符のような検索を可能にします。たとえば、**cn~=first_name last_name**、**cn~=first_name**、または **cn~=first_nam** (スペルに注意) は、**cn=first_name X last_name** というエントリーが返されます。Directory Server の metaphone 表音アルゴリズムは US-ASCII 文字のみをサポートします。したがって、近似インデックスは、英語の値でのみ使用してください。
- 部分文字列インデックス (**sub**) は、維持するためのコストがかかるインデックスですが、エントリー内の部分文字列に対して効率的な検索が可能になります。部分文字列のインデックスは、各エントリーの最小 3 文字に制限されます。たとえば、**telephoneNumber=*555*** と検索すると、**telephoneNumber** 属性に **555** を含む値を持つディレクトリー内のすべてのエントリーが返されます。
- 国際インデックスは、国際ディレクトリー内の情報の検索を迅速化します。国際インデックスの作成プロセスは、通常のインデックスを作成するプロセスと似ています。ただし、オブジェクト識別子 (OID) をインデックス化する属性に関連付けることで一致するルールを適用する点が異なります。

2.2. インデックスのメリットとのバランス

新しいインデックスを作成する前に、インデックスを維持することのメリットとコストのバランスを考えてみます。

- 近似インデックスは、通常、数字を含む属性 (電話番号など) には効率的ではありません。
- 部分文字列のインデックスはバイナリー属性では機能しません。
- イメージなど、大きな値を含む属性の等価インデックスは回避してください。
- 検索で一般的に使用されない属性のインデックスを維持すると、検索パフォーマンスを向上させることなくオーバーヘッドが増加します。

- インデックス化されていない属性は、検索要求で依然として使用できますが、検索のタイプによっては検索パフォーマンスが大幅に低下する可能性があります。

インデックスは、非常に時間がかかります。たとえば、Directory Server が追加操作を受け取った場合、Directory Server はインデックス属性を調べて、属性値に対してインデックスが維持されているかどうかを判断します。作成された属性値がインデックス化されている場合、Directory Server は新しい属性値をインデックスに追加し、その後、実際の属性値がエントリーに作成されます。

例2.1 ユーザーがエントリーを追加する際に Directory Server が行うインデックス作成手順

Directory Server が以下のインデックスを維持していると仮定します。

- **cn** および **sn** 属性の等値性、近似性、および部分文字列インデックス。
- **telephoneNumber** 属性の等価および部分文字列のインデックス。
- **description** 属性の部分文字列インデックス。

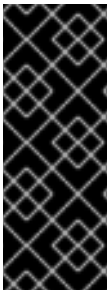
たとえば、ユーザーが次のようなエントリーを追加したとします。

```
dn: cn=John Doe,ou=People,dc=example,dc=com
objectclass: top
objectClass: person
objectClass: orgperson
objectClass: inetorgperson
cn: John Doe
cn: John
sn: Doe
ou: Manufacturing
ou: people
telephoneNumber: 408 555 8834
description: Manufacturing lead
```

ユーザーがエントリーを追加すると、Directory Server は以下のステップを実行します。

1. **John** および **John Doe** の **cn** 等価インデックスエントリーを作成します。
2. **John** および **John Doe** の適切な **cn** の近似インデックスエントリーを作成します。
3. **John** および **John Doe** の **cn** 部分文字列インデックスエントリーを作成します。
4. **Doe** の **sn** 等価インデックスエントリーを作成します。
5. **Doe** に対する **sn** 近似インデックスエントリーを作成します。
6. **Doe** の **sn** 部分文字列インデックスエントリーを作成します。
7. **408 555 8834** の **telephoneNumber** 等価インデックスエントリーを作成します。
8. **408 555 8834** の **telephoneNumber** 部分文字列インデックスエントリーを作成します。
9. **Manufacturing lead** の **description** 部分文字列インデックスエントリーを作成します。

この例は、大規模なディレクトリーのデータベースを作成および保守するために必要なアクションの数が、非常に多くのリソースを消費する可能性があることを示しています。



重要

Directory Server のパフォーマンスに影響を与える可能性があるため、メンバーシップ属性 (**member**、**uniquemember** など) の部分文字列インデックスを定義しないでください。メンバーを追加または削除する場合 (たとえば、多数のメンバーを持つグループに **uniquemember** を追加する場合)、**uniquemember** の部分文字列インデックスの計算では、追加または削除された値だけでなく、すべての **uniquemember** 値を評価する必要があります。

2.3. デフォルトのインデックス属性

Directory Server は、デフォルトのインデックス属性を、**cn=default indexes,cn=config,cn=ldbm database,cn=plugins,cn=config** エントリーに格納します。インデックスタイプを含めて表示するには、次のように入力します。

```
# ldapsearch -D "cn=Directory Manager" -W -H ldap://server.example.com -b "cn=default indexes,cn=config,cn=ldbm database,cn=plugins,cn=config" -s one -o ldif-wrap=no
```

表2.1 Directory Server のデフォルトインデックス属性

aci	**cn**	**entryUSN**
entryUUID	**givenName**	**mail**
mailAlternateAddress	**mailHost**	**member**
memberOf	**nsUniqueId**	**nsCertSubjectDN**
nsTombstoneCSN	**ntUniqueId**	**ntUserDomainId**
numSubordinates	**objectClass**	**owner**
parentId	**seeAlso**	**sn**
targetUniqueId	**telephoneNumber**	**uid**
uniqueMember		

警告

表に記載されている属性 (システムインデックス) をデータベースのインデックスから削除すると、Directory Server のパフォーマンスに大きな影響を与える可能性があります。

2.4. コマンドラインを使用した特定のデータベースのインデックスの維持

dsconf ユーティリティーを使用して、コマンドラインでインデックスの設定を維持することができます。

手順

- たとえば、**userRoot** データベースのインデックスに **roomNumber** 属性を追加するには、インデックスタイプが **eq** と **sub** の場合、次のように入力します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com backend index add --attr roomNumber --index-type eq --index-type sub --reindex userRoot
```

  --reindex オプションを付けると、Directory Server が自動的にデータベースのインデックスを再作成します。

- たとえば、**userRoot** データベースの **roomNumber** 属性のインデックス設定に **pres** インデックスタイプを追加するには、次のように入力します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com backend index set --attr roomNumber --add-type pres userRoot
```

- たとえば、**userRoot** データベースの **roomNumber** 属性のインデックス設定から **pres** インデックスタイプを削除するには、次のように入力します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com backend index set --attr roomNumber --del-type pres userRoot
```

- たとえば、**userRoot** データベースのインデックスから **roomNumber** 属性を削除するには、次のように入力します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com backend index delete --attr roomNumber userRoot
```

検証

- **userRoot** データベースのインデックス設定をリスト表示します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com backend index list userRoot
```

2.5. インスタンスのオフライン時におけるインデックスの再作成

インスタンスのオフライン時に、**dsctl db2index** ユーティリティーを使用してデータベース全体のインデックスを再作成できます。

前提条件

- インデックスエントリーを作成しているか、既存の **userRoot** データベースに追加のインデックスタイプを追加している。

手順

1. インスタンスをシャットダウンします。

```
# dsctl instance_name stop
```

2. インデックスを再作成します。

   a. データベース内のすべてのインデックスに対して、以下を実行します。

```
# dsctl instance_name db2index

[23/Feb/2023:05:38:28.034826108 -0500] - INFO - check_and_set_import_cache - pagesize: 4096, available bytes 1384095744, process usage 27467776
[23/Feb/2023:05:38:28.037952026 -0500] - INFO - check_and_set_import_cache - Import allocates 540662KB import cache.
[23/Feb/2023:05:38:28.055104135 -0500] - INFO - bdb_db2index - userroot: Indexing attribute: aci
...
[23/Feb/2023:05:38:28.134350191 -0500] - INFO - bdb_db2index - userroot: Finished indexing.
[23/Feb/2023:05:38:28.151907852 -0500] - INFO - bdb_pre_close - All database threads now stopped
db2index successful
```

   b. 特定の属性インデックスの場合は、以下を実行します。

```
# dsctl instance_name db2index userRoot --attr aci cn givenname
```

   次のコマンドは、**aci**、**cn**、および **givenname** 属性のインデックスを再作成します。

   c. **dsctl** (オフライン) コマンドの詳細は、次を実行してください。

```
# dsctl instance_name db2index --help
```

3. インスタンスを起動します。

```
# dsctl instance_name start
```

検証

- **userRoot** データベースのインデックス設定をリスト表示します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com backend index list userRoot
```

2.6. WEB コンソールを使用した特定のデータベースのインデックスの維持

Web コンソールを使用して、Directory Server のインデックス設定を維持することができます。

前提条件

- Web コンソールでインスタンスにログインしている。

手順

- **Database → Suffixes → suffix_name → Indexes → Database Indexes** に移動します。

- インデックスに属性を追加するには、以下を行います。
  - **Add Index** をクリックします。
  - **Select An Attribute** フィールドに属性名を入力します。
  - インデックスタイプを選択します。
  - **Index attribute after creation** を選択します。
  - **Create Index** をクリックします。
- 属性のインデックス設定を更新するには、以下を行います。
  - 属性の横にあるオーバーフローメニューをクリックし、**Edit Index** を選択します。
  - インデックスの設定を必要に応じて変更してください。
  - **Index attribute after creation** を選択します。
  - **Save Index** をクリックします。
- インデックスから属性を削除するには、以下を行います。
  - 属性の横にあるオーバーフローメニューをクリックし、**Delete Index** を選択します。
  - **Yes, I am sure** を選択して、**Delete** をクリックします。
  - Suffix Tasks メニューで **Reindex Suffix** を選択します。

検証

- Database → Suffixes → suffix_name → Indexes → Database Indexes に移動し、インデックスの設定に変更が反映されていることを確認します。

第3章 部分文字列インデックスでの検索キーの長さの変更

デフォルトでは、部分文字列インデックスの検索キーの長さは 3 文字以上にする必要があります。たとえば、ディレクトリーサーバーは文字列 **abc** を検索キーとしてインデックスに追加しますが、**ab*** は追加しません。ただし、特に多くのワイルドカード文字を含む検索のパフォーマンスを向上させるために、検索キーの長さを短くすることができます。これにより、インデックス内の検索キーの数が増加します。

ディレクトリーサーバーには、検索キーに必要な最小文字数を変更する 3 つの属性があります。

- **nsSubStrBegin**: 検索キーの先頭 (ワイルドカード文字の前) の最小文字数を設定します。以下に例を示します。

```
abc*
```

- **nsSubStrMiddle**: 検索キーのワイルドカード文字間の最小文字数を設定します。以下に例を示します。

```
*abc*
```

- **nsSubStrEnd**: ワイルドカード文字の後の検索キーの末尾の文字数を設定します。以下に例を示します。

```
*xyz
```

3.1. コマンドラインを使用して部分文字列インデックスの検索キーの長さを変更する

属性インデックスに新しい検索キーの長さを設定することで、検索速度を向上させることができます。

手順

1. 新しい検索キーの長さを設定するには、**extensibleObject** オブジェクトクラスを追加してから、**nsSubStrBegin**、**nsSubStrEnd**、または **nsSubStrMiddle** 属性をエントリーに追加します。以下に例を示します。

```
# ldapmodify -D "cn=Directory Manager" -W -p 389 -h server.example.com -x

dn: attribute_name,cn=index,cn=database_name,cn=ldbm database,cn=plugins,cn=config
changetype: modify
add: objectclass
objectclass: extensibleObject
-
add: nsSubStrBegin
nsSubStrBegin: 2
-
add: nsSubStrMiddle
nsSubStrMiddle: 2
-
add: nsSubStrEnd
nsSubStrEnd: 2
```

2. 新しい設定を適用するには、インデックスを再作成します。たとえば、Directory Server インスタンスの実行中に、次のコマンドを使用して、指定した属性のインデックスを再作成します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com backend index reindex --attr attribute_name database_name
```

検証

- **cn** など、検索キーの長さを変更する属性を選択します。
- **cn** インデックスをダンプします。

```
dbscan -D bdb -f /var/lib/dirsrv/slapd-instance/db/database/cn.db > /tmp/default_len
```

- セクションコマンドラインを使用して部分文字列インデックスの検索キーの長さを変更する の説明に従って、新しい検索キーの長さを設定します。
- インスタンスを停止して、ディスク上のデータベースを同期します。

```
# dsctl instance_name stop
```

- **cn** インデックスをダンプします。

```
dbscan -D bdb -f /var/lib/dirsrv/slapd-instance/db/database/cn.db > /tmp/len_2
```

- **len_2** と **default_len** ファイルを比較します。

```
diff /tmp/len_2 /tmp/default_len
```

第4章 仮想リストビューコントロールを使用して、大規模な検索結果の連続したサブセットを要求する

Directory Server は、LDAP 仮想リストビューコントロールをサポートしています。この制御により、LDAP クライアントは大規模な検索結果の連続したサブセットを要求できます。

たとえば、Directory Server に 100.000 エントリーのアドレス帳を保存したとします。デフォルトでは、すべてのエントリーのクエリーはすべてのエントリーを一度に返します。これはリソースと時間のかかる操作であり、ユーザーが結果をスクロールすると一部のセットしか表示されないため、クライアントはデータセット全体を必要としないことがよくあります。

ただし、クライアントが VLV コントロールを使用する場合、サーバーはサブセットのみを返します。たとえば、ユーザーがクライアントアプリケーションでスクロールすると、サーバーはさらに多くのエントリーを返します。これにより、サーバーの負荷が軽減され、クライアントはすべてのデータを一度に保存して処理する必要がなくなります。

すべての検索パラメーターが固定されている場合、VLV はサーバーでソートされた検索のパフォーマンスも向上させます。Directory Server は、VLV インデックス内の検索結果を事前に計算します。したがって、VLV インデックスは、結果を取得してから後で並べ替えるよりもはるかに効率的です。

Directory Server では、VLV コントロールは常に利用可能です。ただし、大きなディレクトリーで使用する場合は、ブラウジングインデックスとも呼ばれる VLV インデックスを使用すると、速度が大幅に向上します。

Directory Server は、標準インデックスなどの属性の VLV インデックスを維持しません。サーバーは、エントリーに設定された属性とディレクトリーツリー内のそれらのエントリーの場所に基づいて、VLV インデックスを動的に生成します。標準エントリーとは異なり、VLV エントリーはデータベース内の特別なエントリーです。

4.1. LDAPSEARCH コマンドでの VLV コントロールの動作

通常、LDAP クライアントアプリケーションでは仮想リストビュー (VLV) 機能を使用します。ただし、たとえばテスト目的で、**ldapsearch** ユーティリティーを使用して部分的な結果のみを要求できます。

ldapsearch コマンドで VLV 機能を使用するには、**sss** (サーバー側の並べ替え) と **vlv** 検索拡張機能の両方に **-E** オプションを指定します。

```
# ldapsearch ... -E 'sss=attribute_list' -E 'vlv=query_options'
```

sss 検索拡張機能の構文は次のとおりです。

```
[!]sss=[-]<attr[:OID]>[/[-]<attr[:OID]>...]
```

vlv 検索拡張機能の構文は次のとおりです。

```
[!]vlv=<before>/<after>(/<offset>/<count>|:<value>)
```

- **before** は、対象のエントリーの前に返されるエントリーの数を設定します。
- **after** は、対象のエントリーの後に返されるエントリーの数を設定します。
- **index**、**count**、および **value** は、ターゲットエントリーの決定に役立ちます。**value** を設定すると、ターゲットエントリーは、その値で始まる最初の並べ替え属性を持つ最初のエントリーになります。それ以外の場合は、**count** を **0** に設定し、ターゲットエントリーは **index** 値 (1 か

ら開始) によって決定されます。**count** 値が **0** より大きい場合、ターゲットエントリーは **index * number of entries / count** によって決定されます。

例4.1 VLV 検索拡張機能を使用した ldapsearch コマンドの出力

次のコマンドは、**ou=People,dc=example,dc=com** を検索します。次に、サーバーは結果を **cn** 属性でソートし、70 番目のエントリーの **uid** 属性を、オフセットの前の 1 つのエントリーと後の 2 つのエントリーとともに返します。

```
# ldapsearch -D "cn=Directory Manager" -W -H ldap://server.example.com -b
"ou=People,dc=example,dc=com" -s one -x -E 'sss=cn' -E 'vlv=1/2/70/0' uid
# user069, People, example.com
dn: uid=user069,ou=People,dc=example,dc=com
uid: user069

# user070, People, example.com
dn: uid=user070,ou=People,dc=example,dc=com
uid: user070

# user071, People, example.com
dn: uid=user071,ou=People,dc=example,dc=com
uid: user071

# user072, People, example.com
dn: uid=user072,ou=People,dc=example,dc=com
uid: user072

# search result
search: 2
result: 0 Success
control: 1.2.840.113556.1.4.474 false MIQAAAADCgEA
sortResult: (0) Success
control: 2.16.840.1.113730.3.4.10 false MIQAAAALAgFGAgMAnaQKAQA=
vlvResult: pos=70 count=40356 context= (0) Success

# numResponses: 5
# numEntries: 4
Press [before/after(/offset/count|:value)] Enter for the next window.
```

関連情報

ldapsearch(1) man ページの **-E** パラメーターの説明。

4.2. 認証されていないユーザーが VLV コントロールを使用できるようにする

デフォルトでは、**oid=2.16.840.1.113730.3.4.9,cn=features,cn=config** エントリーのアクセス制御命令 (ACI) により、認証されたユーザーのみが VLV コントロールを使用できるようになります。認証されていないユーザーも VLV コントロールを使用できるようにするには、**userdn = "ldap:///all"** を **userdn = "ldap:///anyone"** に変更して ACI を更新します。

手順

- **oid=2.16.840.1.113730.3.4.9,cn=features,cn=config** の ACI を更新します。

```
# ldapmodify -D "cn=Directory Manager" -W -H ldap://server.example.com -x

dn: oid=2.16.840.1.113730.3.4.9,cn=features,cn=config
changetype: modify
replace: aci
aci: (targetattr != "aci")(version 3.0; acl "VLV Request Control"; allow( read, search,
compare, proxy ) userdn = "ldap:///anyone";)
```

検証

- バインドユーザーを指定せずに、VLV コントロールでクエリーを実行します。

```
# ldapsearch -H ldap://server.example.com -b "ou=People,dc=example,dc=com" -s one
-x -E 'sss=cn' -E 'vlv=1/2/70/0' uid
```

このコマンドでは、サーバーが匿名バインドを許可する必要があります。

コマンドが成功してもエントリーが返されない場合は、バインドユーザーを使用してクエリーを再度実行し、認証の使用時にクエリーが機能することを確認します。

関連情報

- 匿名バインドの無効化

4.3. コマンドラインを使用して VLV インデックスを作成し、VLV クエリーの速度を向上させる

この手順に従って、**mail** 属性を含み、**objectClass** 属性が **person** に設定されている **ou=People,dc=example,dc=com** 内のエントリーに対して、ブラウジングインデックスとも呼ばれる仮想リストビュー (VLV) インデックスを作成します。

前提条件

- クライアントアプリケーションは VLV コントロールを使用している。
- クライアントアプリケーションでは、大規模な検索結果の連続したサブセットに対してクエリーを実行する必要がある。
- ディレクトリーには多数のエントリーが含まれている。

手順

1. VLV 検索エントリーを作成します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com backend vlv-index
add-search --name "VLV People" --search-base "ou=People,dc=example,dc=com" --
search-filter "(&(objectClass=person)(mail=*))" --search-scope 2 userRoot
```

コマンドは、以下のオプションを使用します。

- **--name** は検索エントリーの名前を設定します。これは任意の名前にすることができます。

- **--search-base** は、VLV インデックスのベース DN を設定します。Directory Server は、このエントリーに VLV インデックスを作成します。
- **--search-scope** は、VLV インデックス内のエントリーに対して実行する検索の範囲を設定します。このオプションは、**0** (ベース検索)、**1** (1 レベル検索)、または **2** (サブツリー検索) に設定できます。
- **--search-filter** は、ディレクトリーサーバーが VLV インデックスを作成するときに適用するフィルターを設定します。このフィルターに一致するエントリーのみがインデックスの一部になります。
- **userRoot** は、エントリーを作成するデータベースの名前です。

2. インデックスエントリーを作成します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com backend vlv-index
add-index --index-name "VLV People - cn sn" --parent-name "VLV People" --sort "cn
sn" --index-it userRoot
```

コマンドは、以下のオプションを使用します。

- **--index-name** は、インデックスエントリーの名前を設定します。これは任意の名前にすることができます。
- **--parent-name** は、VLV 検索エントリーの名前を設定し、前の手順で設定した名前と一致する必要があります。
- **--sort** は属性名とソート順序を設定します。属性をスペースで区切ります。
- **--index-it** は、エントリーの作成後に、ディレクトリーサーバーがインデックスタスクを自動的に開始します。
- **userRoot** は、エントリーを作成するデータベースの名前です。

検証

1. **/var/log/dirsrv/slapd-instance_name/errors** ファイルで VLV インデックスが正常に作成されたことを確認します。

```
[26/Nov/2021:11:32:59.001988040 +0100] - INFO - bdb_db2index - userroot: Indexing VLV:
VLV People - cn sn
[26/Nov/2021:11:32:59.507092414 +0100] - INFO - bdb_db2index - userroot: Indexed 1000
entries (2%).
...
[26/Nov/2021:11:33:21.450916820 +0100] - INFO - bdb_db2index - userroot: Indexed 40000
entries (98%).
[26/Nov/2021:11:33:21.671564324 +0100] - INFO - bdb_db2index - userroot: Finished
indexing.
```

2. **ldapsearch** コマンドで VLV コントロールを使用して、ディレクトリーから特定のレコードのみを照会します。

```
# ldapsearch -D "cn=Directory Manager" -W -H ldap://server.example.com -b
"ou=People,dc=example,dc=com" -s one -x -E 'sss=cn' -E 'vlv=1/2/70/0' uid
# user069, People, example.com
dn: uid=user069,ou=People,dc=example,dc=com
```

```
cn: user069

# user070, People, example.com
dn: uid=user070,ou=People,dc=example,dc=com
cn: user070

# user071, People, example.com
dn: uid=user071,ou=People,dc=example,dc=com
cn: user071

# user072, People, example.com
dn: uid=user072,ou=People,dc=example,dc=com
cn: user072
```

この例では、**ou=People,dc=example,dc=com** に**uid=user001** から少なくとも **uid=user072** まで連続して名前が付けられたエントリーがあることを前提としています。

関連情報

- **ldapsearch(1)** man ページの **-E** パラメーターの説明。
- ldapsearch コマンドの VLV コントロール

4.4. WEB コンソールを使用して VLV インデックスを作成し、VLV クエリーの速度を向上させる

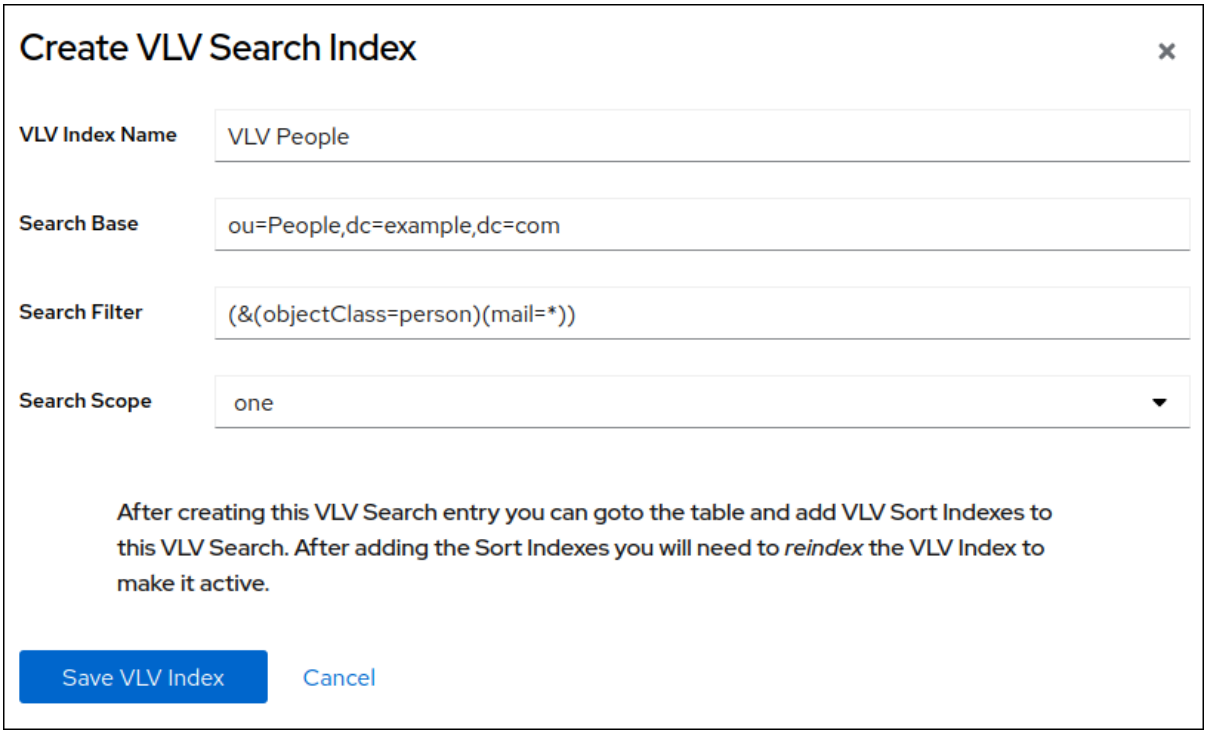
この手順に従って、**mail** 属性を含み、**objectClass** 属性が **person** に設定されている **ou=People,dc=example,dc=com** 内のエントリーに対して、ブラウジングインデックスとも呼ばれる仮想リストビュー (VLV) インデックスを作成します。

前提条件

- Web コンソールでインスタンスにログインしている。
- クライアントアプリケーションは VLV コントロールを使用している。
- クライアントアプリケーションでは、大規模な検索結果の連続したサブセットに対してクエリーを実行する必要がある。
- ディレクトリーには多数のエントリーが含まれている。

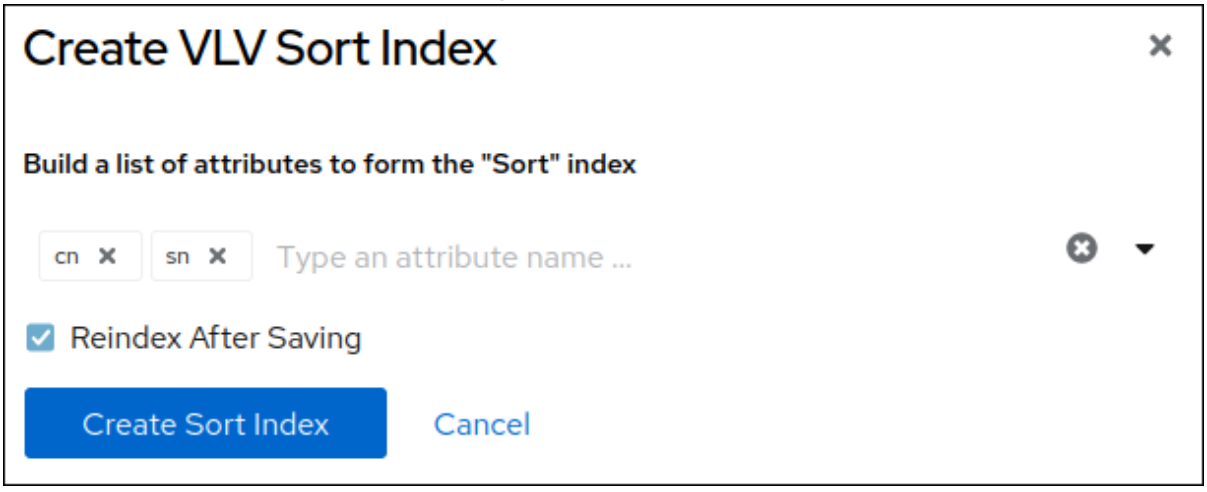
手順

1. **Database** → **Suffixes** → **dc=example,dc=com** → **VLV Indexes** に移動します。
2. **Create VLV Index** をクリックして、フィールドに入力します。



- **VLV Index Name**: 検索エントリーの名前です。これは任意の名前にすることができます。
- **Search base**: VLV インデックスのベース DN です。Directory Server は、このエントリーに VLV インデックスを作成します。
- **Search Filter**: ディレクトリーサーバーが VLV インデックスを作成するときに適用されるフィルターです。このフィルターに一致するエントリーのみがインデックスの一部になります。
- **Search Scope**: VLV インデックス内のエントリーに対して実行する検索の範囲です。

3. **Save VLV Index** をクリックします。
4. **Create Sort Index** をクリックします。
5. 属性名を入力し、**Reindex After Saving** を選択します。



6. **Create Sort Index** をクリックします。

検証

1. **Monitoring → Logging → Errors Log** に移動し、VLV インデックスが正常に作成されたことを確認します。

```
[26/Nov/2021:11:32:59.001988040 +0100] - INFO - bdb_db2index - userroot: Indexing VLV: VLV People - cn sn
[26/Nov/2021:11:32:59.507092414 +0100] - INFO - bdb_db2index - userroot: Indexed 1000 entries (2%).
...
[26/Nov/2021:11:33:21.450916820 +0100] - INFO - bdb_db2index - userroot: Indexed 40000 entries (98%).
[26/Nov/2021:11:33:21.671564324 +0100] - INFO - bdb_db2index - userroot: Finished indexing.
```

2. **ldapsearch** コマンドで VLV コントロールを使用して、ディレクトリーから特定のレコードのみを照会します。

```
# ldapsearch -D "cn=Directory Manager" -W -H ldap://server.example.com -b "ou=People,dc=example,dc=com" -s one -x -E 'sss=cn' -E 'vlv=1/2/70/0' uid
# user069, People, example.com
dn: uid=user069,ou=People,dc=example,dc=com
cn: user069

# user070, People, example.com
dn: uid=user070,ou=People,dc=example,dc=com
cn: user070

# user071, People, example.com
dn: uid=user071,ou=People,dc=example,dc=com
cn: user071

# user072, People, example.com
dn: uid=user072,ou=People,dc=example,dc=com
cn: user072
```

この例では、**ou=People,dc=example,dc=com** に**uid=user001** から少なくとも **uid=user072** まで連続して名前が付けられたエントリーがあることを前提としています。

関連情報

- **ldapsearch(1)** man ページの **-E** パラメーターの説明。
- ldapsearch コマンドの VLV コントロール