Red Hat Directory Server 12

인덱스 관리

인덱스를 최적화하여 검색 성능 개선

Last Updated: 2024-08-25

Red Hat Directory Server 12 인덱스 관리

인덱스를 최적화하여 검색 성능 개선

법적 공지

초록

인덱싱을 사용하면 특성 또는 값을 분류하고 분류하여 정보를 더 빠르게 검색하고 검색할 수 있습니다. 가상 목록 보기 컨트롤을 사용하여 대규모 검색 결과의 연속적인 하위 집합을 요청할 수 있습니다.

차례

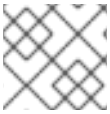
RED HAT DIRECTORY SERVER에 대한 피드백 제공

Red Hat의 문서 및 제품에 대한 의견을 제공해 주셔서 감사합니다. Red Hat이 어떻게 이를 개선할 수 있는지 알려 주십시오. 이렇게 하려면 다음을 수행합니다.

- Jira (계정 필요)를 통해 Red Hat Directory Server 설명서에 피드백을 제출하려면 다음을 수행합니다.
  1. Red Hat 문제 추적기 로 이동하십시오.
  2. **요약** 필드에 설명 제목을 입력합니다.
  3. **설명** 필드에 개선을 위한 제안을 입력합니다. 문서의 관련 부분에 대한 링크를 포함합니다.
  4. 대화 상자 하단에서 **생성** 을 클릭합니다.
- Jira를 통해 Red Hat Directory Server 제품에 대한 피드백을 제출하기 위해 필요한 경우:
  1. Red Hat 문제 추적기 로 이동하십시오.
  2. **문제 생성** 페이지에서 **다음을** 클릭합니다.
  3. **Summary** 필드를 입력합니다.
  4. **Component** 필드에서 구성 요소를 선택합니다.
  5. 다음을 포함하여 **Description** 필드를 작성합니다.
     a. 선택한 구성 요소의 버전 번호입니다.
     b. 문제 또는 개선을 위한 제안을 재현하는 단계입니다.
  6. **생성**을 클릭합니다.

1장. 새로 생성된 모든 데이터베이스에 적용되는 기본 인덱스 정의

Directory Server의 기본 인덱스는 인덱싱할 특성 집합을 정의합니다. 새 데이터베이스를 만들 때 Directory Server는 기본 인덱스 특성을 **cn=default 인덱스,cn=config,cn=plugins,cn=plugins,cn=plugins,cn=config** 항목에서 데이터베이스 특정 cn=index, **cn=index,cn=*name*,cn=plugins,cn=plugins,cn=config** 항목에 복사합니다.



참고

Directory Server는 기본 인덱스의 변경 사항을 기존 데이터베이스에 적용하지 않습니다.

1.1. 다양한 인덱스 유형

Directory Server는 인스턴스의 데이터베이스 디렉터리에 있는 별도의 데이터베이스 파일에 인덱싱된 각 특성의 인덱스를 저장합니다. 예를 들어 **sn** 속성의 인덱스는 **/var/lib/dirsrv/slapd-*instance_name*/db/*name*/sn.db** 파일에 저장됩니다. 각 인덱스 파일은 Directory Server가 특성에 대해 서로 다른 인덱스를 유지 관리하는 경우 여러 인덱스 유형을 포함할 수 있습니다.

Directory Server는 다음과 같은 인덱스 유형을 지원합니다.

- 표시 인덱스(**pres**)는 특정 특성을 포함하는 항목 목록입니다. 예를 들어 클라이언트가 **속성=mail** 과 같이 검색을 자주 수행할 때 이 유형을 사용합니다.
- 같음 인덱스(**eq**)는 특정 특성 값을 포함하는 항목을 검색합니다. 예를 들어 **cn** 특성의 같음 인덱스를 사용하면 **cn=*first_name last_name*** 을 더 빠르게 검색할 수 있습니다.
- 대략적인 인덱스(**Approx**)를 사용하면 대략적인 또는 난증과 같은 검색을 효율적으로 수행할 수 있습니다. 예를 들어, **cn~=*first_name last_name***, **cn~=*first_name***, **cn~=*first_nam*** 을 검색하면 **cn=*first_name X last_name*** 을 반환합니다. Directory Server의 meta phone phonetic 알고리즘은 US-ASCII 문자만 지원합니다. 따라서 대략적인 인덱싱을 영어 값과 함께 사용해야 합니다.
- 하위 문자열 인덱스(**sub**)는 유지 관리할 수 있는 비용이 많이 드는 인덱스이지만 항목 내의 하위 문자열에 대해 효율적으로 검색할 수 있습니다. 하위 문자열 인덱스는 각 항목에 대해 최소 3자로 제한됩니다. 예를 들어, phone **Number=*555*** 를 검색하면 디렉토리의 모든 항목이 phone **Number** 속성에 **555** 를 포함하는 값을 반환합니다.
- 국제 인덱스를 사용하면 검색 속도를 높일 수 있으므로 국제 디렉터리에서 정보를 확인할 수 있습니다. 국제 인덱스를 생성하는 프로세스는 인덱스할 특성과 OID(Object identifier)를 연결하여 일치하는 규칙을 적용하는 것을 제외하고 일반 인덱스를 생성하는 프로세스와 유사합니다.

1.2. 인덱싱의 이점 밸런싱

새 인덱스를 만들기 전에 인덱스를 유지 관리하는 이점과 비용의 균형을 유지합니다.Before you create new indexes, balance the benefits of maintaining indexes against the costs:

- 대략적인 인덱스는 전화 번호와 같이 숫자를 일반적으로 포함하는 속성에는 효율적이지 않습니다.
- 바이너리 특성에는 하위 문자열 인덱스가 작동하지 않습니다.
- 이미지와 같이 큰 값이 포함된 특성에 대한 같음 인덱스를 방지할 수 있습니다.
- 검색에 일반적으로 사용되지 않는 속성의 인덱스를 유지 관리하면 검색 성능을 향상시키지 않고 오버헤드가 증가합니다.

- 검색 유형에 따라 검색 성능이 크게 저하될 수 있지만 인덱싱되지 않은 속성은 검색 요청에 계속 사용될 수 있습니다.

인덱스에는 시간이 많이 걸릴 수 있습니다. 예를 들어 Directory Server에서 추가 작업을 수신하면 서버는 인덱싱 특성을 검사하여 특성 값에 대해 인덱스가 유지 관리되는지 확인합니다. 생성된 특성 값이 인덱싱되면 Directory Server에서 새 특성 값을 인덱스에 추가한 다음 실제 특성 값이 해당 항목에 생성됩니다.

예 1.1. 사용자가 항목을 추가할 때 Directory Server가 수행하는 인덱싱 단계

Directory Server에서 다음과 같은 인덱스를 유지 관리하는 것으로 가정합니다.

- **cn** 및 **sn** 속성에 대한 같음, 대략적, 하위 문자열 인덱스입니다.
- phone **Number** 속성에 대한 같음 및 하위 문자열 인덱스입니다.
- **description** 속성에 대한 하위 문자열 인덱스입니다.

예를 들어 사용자는 다음 항목을 추가합니다.

```
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
```

사용자가 항목을 추가하면 Directory Server는 다음 단계를 수행합니다.

1. **존과 존 Doe** 에 대한 **cn** equality 인덱스 항목을 만듭니다.
2. **존과 존 Doe** 에 대한 **cn** 대략 인덱스 항목을 만듭니다.
3. **존과 존 Doe** 에 대한 **cn** 하위 문자열 인덱스 항목을 만듭니다.
4. **Doe** 에 대한 **sn** equality 인덱스 항목을 만듭니다.
5. **Doe** 에 대한 **sn** 대략 인덱스 항목을 만듭니다.
6. **Doe** 에 대한 **sn** 하위 문자열 인덱스 항목을 만듭니다.
7. **408 555 8834** 에 대한 phone **Number** 같음 인덱스 항목을 만듭니다.
8. **408 555 8834** 에 대한 phone **Number** 하위 문자열 인덱스 항목을 만듭니다.
9. **Manufacturing lead** 에 대한 **description** substring index 항목을 생성합니다.

이 예제에서는 대규모 디렉터리에 대한 데이터베이스를 만들고 유지 관리하는 데 필요한 작업 수가 리소스 집약적일 수 있음을 보여줍니다.

중요

Directory Server 성능에 영향을 줄 수 있으므로 멤버십 속성(예: 멤버 ,**unique member**)에 대한 하위 문자열 인덱스를 정의하지 마십시오. 멤버를 추가하거나 제거할 때,**uniquemember** 가 많은 멤버가 있는 그룹에 uniquemember를 추가 또는 제거할 때 **uniquemember** 하위 문자열 인덱스의 계산에는 추가 또는 제거된 값뿐만 아니라 모든 **uniquemember** 값을 평가해야 합니다.

1.3. 기본 인덱스 속성

디렉터리 서버는 기본 인덱스 속성을 **cn=default 인덱스,cn=config,cn=ldbm 데이터베이스,cn=plugins,cn=config** 항목에 저장합니다. 인덱스 유형을 포함하여 이를 표시하려면 다음을 입력합니다.

```
# 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
```

표 1.1. Directory Server 기본 인덱스 특성

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

주의

데이터베이스 인덱스에서 테이블(시스템 인덱스)에 나열된 특성을 제거하면 Directory Server 성능에 큰 영향을 미칠 수 있습니다.

1.4. 기본 인덱스 유지 관리

디렉터리 서버는 기본 인덱스 속성을 **cn=default 인덱스,cn=config,cn=ldbm 데이터베이스,cn=plugins,cn=config** 항목에 저장합니다. LDIF 문을 사용하여 기본 인덱스 특성만 유지 관리할 수 있습니다.

절차

- 예를 들어 index 유형이 **eq** 및 **sub** 인인 기본 인덱스에 **roomNumber** 특성을 추가하려면 다음을 입력합니다.

```
# 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
```

  LDIF 설명:

  - **Objectclass: nsIndex**: 이 항목이 인덱스 항목임을 정의합니다.
  - **Objectclass: top**: 이 오브젝트 클래스는 인덱스 항목에서 추가로 필요합니다.
  - **CN**: 특성의 이름을 index로 설정합니다.
  - **nsSystemIndex**: 인덱스가 Directory Server 작업에 필요한지 여부를 나타냅니다.
  - **nsIndexType**: 이 다중 값 속성은 인덱스 유형을 지정합니다.

- 예를 들어, **roomNumber** 속성의 기본 인덱스 속성에 **pres** 인덱스 유형을 추가하려면 다음을 입력합니다.

```
# 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
```

- 예를 들어, **roomNumber** 속성의 기본 인덱스 속성에서 **pres** 인덱스 유형을 제거하려면 다음을 입력합니다.

```
# 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
```

- 예를 들어 기본 인덱스에서 **roomNumber** 속성을 제거하려면 다음을 입력합니다.

```
# 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
```

검증

- 기본 인덱스 특성을 나열하여 변경 사항을 확인합니다.

```
# 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
```

2장. 특정 데이터베이스의 인덱스 유지 관리

Directory Server의 각 데이터베이스에는 자체 인덱스가 있습니다. **dsconf** 유틸리티 또는 웹 콘솔을 사용하여 인덱스를 생성, 업데이트 및 삭제할 수 있습니다.

2.1. 다양한 인덱스 유형

Directory Server는 인스턴스의 데이터베이스 디렉터리에 있는 별도의 데이터베이스 파일에 인덱싱된 각 특성의 인덱스를 저장합니다. 예를 들어 **sn** 속성의 인덱스는 **/var/lib/dirsrv/slapd-*instance_name*/db/*name*/sn.db** 파일에 저장됩니다. 각 인덱스 파일은 Directory Server가 특성에 대해 서로 다른 인덱스를 유지 관리하는 경우 여러 인덱스 유형을 포함할 수 있습니다.

Directory Server는 다음과 같은 인덱스 유형을 지원합니다.

- 표시 인덱스(**pres**)는 특정 특성을 포함하는 항목 목록입니다. 예를 들어 클라이언트가 **속성=mail** 과 같이 검색을 자주 수행할 때 이 유형을 사용합니다.
- 같음 인덱스(**eq**)는 특정 특성 값을 포함하는 항목을 검색합니다. 예를 들어 **cn** 특성의 같음 인덱스를 사용하면 **cn=*first_name last_name*** 을 더 빠르게 검색할 수 있습니다.
- 대략적인 인덱스(**Approx**)를 사용하면 대략적인 또는 난증과 같은 검색을 효율적으로 수행할 수 있습니다. 예를 들어, **cn~=*first_name last_name***, **cn~=*first_name***, **cn~=*first_nam*** 을 검색하면 **cn=*first_name X last_name*** 을 반환합니다. Directory Server의 meta phone phonetic 알고리즘은 US-ASCII 문자만 지원합니다. 따라서 대략적인 인덱싱을 영어 값과 함께 사용해야 합니다.
- 하위 문자열 인덱스(**sub**)는 유지 관리할 수 있는 비용이 많이 드는 인덱스이지만 항목 내의 하위 문자열에 대해 효율적으로 검색할 수 있습니다. 하위 문자열 인덱스는 각 항목에 대해 최소 3자로 제한됩니다. 예를 들어, phone **Number=**555**** 를 검색하면 디렉토리의 모든 항목이 phone **Number** 속성에 **555** 를 포함하는 값을 반환합니다.
- 국제 인덱스를 사용하면 검색 속도를 높일 수 있으므로 국제 디렉터리에서 정보를 확인할 수 있습니다. 국제 인덱스를 생성하는 프로세스는 인덱스할 특성과 OID(Object identifier)를 연결하여 일치하는 규칙을 적용하는 것을 제외하고 일반 인덱스를 생성하는 프로세스와 유사합니다.

2.2. 인덱싱의 이점 밸런싱

새 인덱스를 만들기 전에 인덱스를 유지 관리하는 이점과 비용의 균형을 유지합니다.Before you create new indexes, balance the benefits of maintaining indexes against the costs:

- 대략적인 인덱스는 전화 번호와 같이 숫자를 일반적으로 포함하는 속성에는 효율적이지 않습니다.
- 바이너리 특성에는 하위 문자열 인덱스가 작동하지 않습니다.
- 이미지와 같이 큰 값이 포함된 특성에 대한 같음 인덱스를 방지할 수 있습니다.
- 검색에 일반적으로 사용되지 않는 속성의 인덱스를 유지 관리하면 검색 성능을 향상시키지 않고 오버헤드가 증가합니다.
- 검색 유형에 따라 검색 성능이 크게 저하될 수 있지만 인덱싱되지 않은 속성은 검색 요청에 계속 사용될 수 있습니다.

인덱스에는 시간이 많이 걸릴 수 있습니다. 예를 들어 Directory Server에서 추가 작업을 수신하면 서버는 인덱싱 특성을 검사하여 특성 값에 대해 인덱스가 유지 관리되는지 확인합니다. 생성된 특성 값이 인덱싱되면 Directory Server에서 새 특성 값을 인덱스에 추가한 다음 실제 특성 값이 해당 항목에 생성됩니다.

-

예 2.1. 사용자가 항목을 추가할 때 Directory Server가 수행하는 인덱싱 단계

Directory Server에서 다음과 같은 인덱스를 유지 관리하는 것으로 가정합니다.

- **cn** 및 **sn** 속성에 대한 같음, 대략적, 하위 문자열 인덱스입니다.
- phone **Number** 속성에 대한 같음 및 하위 문자열 인덱스입니다.
- **description** 속성에 대한 하위 문자열 인덱스입니다.

예를 들어 사용자는 다음 항목을 추가합니다.

```
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
```

사용자가 항목을 추가하면 Directory Server는 다음 단계를 수행합니다.

1. **존과 존 Doe** 에 대한 **cn** equality 인덱스 항목을 만듭니다.
2. **존과 존 Doe** 에 대한 **cn** 대략 인덱스 항목을 만듭니다.
3. **존과 존 Doe** 에 대한 **cn** 하위 문자열 인덱스 항목을 만듭니다.
4. **Doe** 에 대한 **sn** equality 인덱스 항목을 만듭니다.
5. **Doe** 에 대한 **sn** 대략 인덱스 항목을 만듭니다.
6. **Doe** 에 대한 **sn** 하위 문자열 인덱스 항목을 만듭니다.
7. **408 555 8834** 에 대한 phone **Number** 같음 인덱스 항목을 만듭니다.
8. **408 555 8834** 에 대한 phone **Number** 하위 문자열 인덱스 항목을 만듭니다.
9. **Manufacturing lead** 에 대한 **description** substring index 항목을 생성합니다.

이 예제에서는 대규모 디렉터리에 대한 데이터베이스를 만들고 유지 관리하는 데 필요한 작업 수가 리소스 집약적일 수 있음을 보여줍니다.

중요

Directory Server 성능에 영향을 줄 수 있으므로 멤버십 속성(예: 멤버 ,**unique member**)에 대한 하위 문자열 인덱스를 정의하지 마십시오. 멤버를 추가하거나 제거할 때,**uniquemember** 가 많은 멤버가 있는 그룹에 uniquemember를 추가 또는 제거할 때 **uniquemember** 하위 문자열 인덱스의 계산에는 추가 또는 제거된 값뿐만 아니라 모든 **uniquemember** 값을 평가해야 합니다.

2.3. 기본 인덱스 속성

디렉터리 서버는 기본 인덱스 속성을 **cn=default 인덱스,cn=config,cn=ldbm 데이터베이스,cn=plugins,cn=config** 항목에 저장합니다. 인덱스 유형을 포함하여 이를 표시하려면 다음을 입력합니다.

```
# 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
```

표 2.1. Directory Server 기본 인덱스 특성

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

주의

데이터베이스 인덱스에서 테이블(시스템 인덱스)에 나열된 특성을 제거하면 Directory Server 성능에 큰 영향을 미칠 수 있습니다.

2.4. 명령줄을 사용하여 특정 데이터베이스의 인덱스 유지 관리

dsconf 유틸리티를 사용하여 명령줄을 사용하여 인덱스 설정을 유지 관리할 수 있습니다.

절차

- 예를 들어, index type **eq** 및 **sub** 를 사용하여 **userRoot** 데이터베이스의 인덱스에 **roomNumber** 특성을 추가하려면 다음을 입력합니다.

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

--reindex 옵션을 사용하면 Directory Server가 데이터베이스를 자동으로 다시 인덱싱합니다.

- 예를 들어 **userRoot** 데이터베이스의 **roomNumber** 속성의 인덱스 설정에 **pres** 인덱스 유형을 추가하려면 다음을 입력합니다.

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

- 예를 들어 **userRoot** 데이터베이스의 **roomNumber** 속성의 인덱스 설정에서 **pres** 인덱스 유형을 제거하려면 다음을 입력합니다.

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

- 예를 들어 **userRoot** 데이터베이스의 인덱스에서 **roomNumber** 특성을 제거하려면 다음을 입력합니다.

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

검증

- **userRoot** 데이터베이스의 인덱스 설정을 나열합니다.

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

2.5. 인스턴스가 오프라인인 동안 인덱스 다시 생성

인스턴스가 오프라인 상태인 동안 **dsctl db2index** 유틸리티를 사용하여 전체 데이터베이스를 다시 인덱싱할 수 있습니다.

사전 요구 사항

- 인덱싱 항목을 생성하거나 기존 **userRoot** 데이터베이스에 추가 인덱스 유형을 추가했습니다.

절차

1. 인스턴스를 종료합니다.

```
# dsctl instance_name stop
```

2. 인덱스를 다시 생성합니다.

   a. 데이터베이스의 모든 인덱스에 대해 다음을 실행합니다.

```
# 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. 특정 특성 인덱스의 경우 다음을 실행합니다.

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

다음 명령은 **aci**,**cn**, **givenname** 특성에 대한 인덱스를 다시 생성합니다.

c. **dsctl** (offline) 명령에 대한 자세한 내용은 다음을 실행합니다.

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

3. 인스턴스를 시작합니다.

```
# dsctl instance_name start
```

검증

- **userRoot** 데이터베이스의 인덱스 설정을 나열합니다.

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

2.6. 웹 콘솔을 사용하여 특정 데이터베이스의 인덱스 유지 관리

웹 콘솔을 사용하여 Directory Server에서 인덱스 설정을 유지 관리할 수 있습니다.

사전 요구 사항

- 웹 콘솔에서 인스턴스에 로그인되어 있습니다.

절차

- Database → Suffixes → *suffix_name* → Indexes → Database Indexes 로 이동합니다.
  - 인덱스에 속성을 추가하려면 다음을 수행합니다.
    - **인덱스 추가를** 클릭합니다.
    - 속성 **선택 필드에 속성** 이름을 입력합니다.
    - 인덱스 유형을 선택합니다.
    - **생성 후 인덱스 특성을** 선택합니다.
    - **인덱스 생성을** 클릭합니다.
  - 특성의 인덱스 설정을 업데이트하려면 다음을 수행합니다.

- 속성 옆에 있는 오버플로 메뉴를 클릭하고 **인덱스 편집** 을 선택합니다.
- 필요에 따라 인덱스 설정을 업데이트합니다.
- **생성 후 인덱스 특성을** 선택합니다.
- **Save Index** 를 클릭합니다.

- 인덱스에서 특성을 삭제하려면 다음을 수행합니다.
  - 속성 옆에 있는 오버플로 메뉴를 클릭하고 **인덱스 삭제** 를 선택합니다.
  - **Yes, I am sure** 를 선택하고 **삭제** 를 클릭합니다.
  - Suffix Tasks 메뉴에서 **Reindex Suffix** 를 선택합니다.

검증

- Database → Suffixes → *suffix_name* → Indexes → Database Indexes 로 이동하여 인덱스 설정에 변경 사항이 반영되는지 확인합니다.

3장. 하위 문자열 인덱스의 검색 키 길이 변경

기본적으로 하위 문자열 인덱스의 검색 키 길이는 3자 이상이어야 합니다.By default, the length of the search key for substring indexes must be at least **three** characters. 예를 들어 Directory Server는 **abc** 문자열을 인덱스에 검색 키로 추가하지만 **ab*** 는 그렇지 않습니다. 그러나 특히 와일드카드 문자를 사용하는 검색의 경우 검색 성능을 개선하기 위해 검색 키 길이를 단축할 수 있습니다. 이렇게 하면 인덱스의 검색 키 수가 증가합니다.

Directory Server에는 검색 키에 필요한 최소 문자 수를 변경하는 세 가지 속성이 있습니다.

- **nsSubStrBegin**: 와일드카드 문자 앞에 검색 키의 시작 부분에 대한 최소 문자 수를 설정합니다. 예를 들면 다음과 같습니다.

```
abc*
```

- **nsSubStrMiddle**: 와일드카드 문자 간에 검색 키의 최소 문자 수를 설정합니다. 예를 들면 다음과 같습니다.

```
*abc*
```

- **nsSubStrEnd**: 와일드카드 문자 뒤에 검색 키 끝에 대한 문자 수를 설정합니다. 예를 들면 다음과 같습니다.

```
*xyz
```

3.1. 명령줄을 사용하여 하위 문자열 인덱스의 검색 키 길이 변경

특성 인덱스에 새 검색 키 길이를 설정하여 검색 속도를 개선할 수 있습니다.

절차

1. 새 검색 키 길이를 설정하려면 **ExtensibleObject** 오브젝트 클래스를 추가한 다음 **nsSubStrBegin**,**nsSubStrEnd**, **nsSubStrMiddle** 속성을 항목에 추가합니다. 예를 들면 다음과 같습니다.

```
# 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. 인덱스를 다시 생성하여 새 설정을 적용합니다. 예를 들어 Directory Server 인스턴스가 실행 중인 동안 다음 명령을 사용하여 지정된 속성에 대한 인덱스를 다시 생성합니다.

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

검증

- 검색 키 길이를 변경할 속성을 선택합니다(예: **cn**).
- **cn** 인덱스를 덤프합니다.

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

- 섹션에 설명된 대로 새 검색 키 길이 를 구성합니다. 명령 줄을 사용하여 하위 문자열 인덱스의 검색 키 길이 변경
- 인스턴스를 중지하여 디스크의 데이터베이스를 동기화합니다.

```
# dsctl instance_name stop
```

- **cn** 인덱스를 덤프합니다.

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

- **len_2** 및 **default_len** 파일을 비교합니다.

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

4장. 가상 목록 보기 제어를 사용하여 대규모 검색 결과의 연속 하위 집합 요청

Directory Server는 LDAP 가상 목록 보기 제어를 지원합니다. 이 제어를 사용하면 LDAP 클라이언트에서 대규모 검색 결과의 연속 하위 집합을 요청할 수 있습니다.

예를 들어, Directory Server에 100.000개의 항목이 있는 주소록을 저장했습니다. 기본적으로 모든 항목의 쿼리는 모든 항목을 한 번에 반환합니다. 이는 리소스 및 시간이 많이 소요되는 작업이며 클라이언트는 사용자가 결과를 스크롤하면 부분 집합만 표시되므로 전체 데이터 집합이 필요하지 않은 경우가 많습니다.

그러나 클라이언트가 VLV 컨트롤을 사용하는 경우 서버는 하위 집합만 반환하고 예를 들어 클라이언트 애플리케이션에서 사용자가 스크롤하는 경우 서버는 더 많은 항목을 반환합니다. 이렇게 하면 서버의 부하가 줄어들고 클라이언트가 모든 데이터를 한 번에 저장하고 처리할 필요가 없습니다.

VLV는 또한 모든 검색 매개변수가 수정될 때 서버 정렬된 검색의 성능을 향상시킵니다. Directory Server는 VLV 인덱스 내에서 검색 결과를 미리 계산합니다. 따라서 VLV 인덱스는 결과를 검색하고 나중에 정렬하는 것보다 훨씬 효율적입니다.

Directory Server에서 VLV 컨트롤은 항상 사용할 수 있습니다. 그러나 대형 디렉터리인 VLV 인덱스에서 이를 사용하는 경우 찾아보기 인덱스라고도 하는 VLV 인덱스의 속도를 크게 향상시킬 수 있습니다.

Directory Server는 표준 인덱스와 같은 특성에 대해 VLV 인덱스를 유지 관리하지 않습니다. 서버는 항목에 설정된 속성과 디렉터리 트리에 있는 항목의 위치에 따라 VLV 인덱스를 동적으로 생성합니다. 표준 항목과 달리 VLV 항목은 데이터베이스의 특수 항목입니다.

4.1. LDAPSEARCH 명령에서 VLV 제어 작동 방식

일반적으로 LDAP 클라이언트 애플리케이션에서 VLV(가상 목록 보기) 기능을 사용합니다. 그러나 예를 들어 테스트를 위해 **ldapsearch** 유틸리티를 사용하여 부분적인 결과만 요청할 수 있습니다.

ldapsearch 명령에서 VLV 기능을 사용하려면 **ss** (서버 측 정렬) 및 **vlv** 검색 확장 모두에 **-E** 옵션을 지정합니다.

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

sss 검색 확장에는 다음 구문이 있습니다.

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

vlv 검색 확장 프로그램에는 다음 구문이 있습니다.

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

- 먼저 대상 항목 앞에 반환된 항목 수를 설정합니다.
- **그런 다음 대상 항목 뒤에 반환되는 항목 수를 설정합니다.**
- **Index,count, value help to determine the target entry. 값을 설정하는 경우 대상 항목은 값으로 시작하는 첫 번째 정렬 특성을 갖는 첫 번째 항목입니다. 그렇지 않으면 count 를 0 으로 설**

정하고 대상 항목은 인덱스 값(부터 시작)에 따라 결정됩니다. 카운트 값이 0 보다 크면 대상 항목은 비율 인덱스 * 항목 수 / 수에 따라 결정됩니다.

예 4.1. VLV 검색 확장자를 사용한 ldapsearch 명령의 출력

다음 명령은 **ou=People,dc=example,dc=com** 에서 검색합니다. 그런 다음 서버는 **cn** 속성으로 결과를 정렬하고 오프셋 이후의 한 항목 및 두 개의 항목이 함께 **70** 번째 항목의 **uid** 속성을 반환합니다.

```
# 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.
```

추가 리소스

ldapsearch(1) 매뉴얼 페이지에 **-E** 매개변수 설명입니다.

4.2. 인증되지 않은 사용자가 VLV 제어를 사용하도록 활성화

기본적으로 **oid=2.16.840.1.113730.3.9,cn=features,cn=config** 항목의 **ACI**(액세스 제어 명령)를 사용하면 인증된 사용자만 **VLV** 제어를 사용할 수 있습니다. 인증되지 않은 사용자도 **VLV** 제어를 사용하도록

활성화하려면 userdn = "ldap:///all" 을 userdn = "ldap:///anyone"로 변경하여 ACI를 업데이트합니다.

절차

- oid=2.16.840.1.113730.3.9,cn=features,cn=config에서 ACI를 업데이트합니다.

```
# 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";)
```

검증

- VLV 컨트롤로 쿼리를 수행해도 바인딩 사용자는 지정할 수 없습니다.

```
# 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
```

이 명령을 사용하려면 서버가 익명의 바인딩을 허용해야 합니다.

명령이 성공하지만 항목을 반환하지 않는 경우 bind 사용자로 쿼리를 다시 실행하여 인증을 사용할 때 쿼리가 작동하는지 확인합니다.

추가 리소스

- 익명 바인딩 비활성화

4.3. 명령줄을 사용하여 VLV 쿼리의 속도를 개선하기 위해 VLV 인덱스 생성

다음 절차에 따라 mail 속성이 포함되어 objectClass 속성을 포함하는 ou=People,dc=example,dc=com 의 항목에 대해 검색 인덱스라고도 하는 가상 목록 보기(VLV) 인덱스를 생성합니다.

사전 요구 사항

- 클라이언트 애플리케이션에서 VLV 제어를 사용합니다.

- 클라이언트 애플리케이션은 대규모 검색 결과의 연속 하위 집합을 쿼리해야 합니다.
- 디렉터리에는 많은 수의 항목이 포함되어 있습니다.

절차

1. VLV 검색 항목을 생성합니다.

```
# 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
```

이 명령은 다음 옵션을 사용합니다.

- --name 은 검색 항목의 이름을 설정합니다. 이것은 어떤 이름일 수 있습니다.
- --search-base 는 VLV 인덱스의 기본 DN을 설정합니다. Directory Server는 이 항목에 VLV 인덱스를 생성합니다.
- --search-scope 는 VLV 인덱스에서 항목을 실행하도록 검색 범위를 설정합니다. 이 옵션을 0 (기본 검색), 1 (한 수준 검색) 또는 2 (subtree 검색)로 설정할 수 있습니다.
- --search-filter 는 VLV 인덱스를 생성할 때 필터 Directory Server가 적용됩니다. 이 필터와 일치하는 항목만 인덱스의 일부가 됩니다.
- userRoot 는 항목을 생성할 데이터베이스의 이름입니다.

2. 인덱스 항목을 생성합니다.

```
# 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
```

이 명령은 다음 옵션을 사용합니다.

- **--index-name** 은 인덱스 항목의 이름을 설정합니다. 이것은 어떤 이름일 수 있습니다.
- **--parent-name** 은 VLV 검색 항목의 이름을 설정하고 이전 단계에서 설정한 이름과 일치해야 합니다.
- **--sort** 는 특성 이름과 정렬 순서를 설정합니다. 특성을 공백으로 구분합니다.
- **--index-it** 로 인해 항목이 생성된 후 **Directory Server**가 인덱스 작업을 자동으로 시작합니다.
- **userRoot** 는 항목을 생성할 데이터베이스의 이름입니다.

검증

1. **/var/log/dirsrv/slapd-*instance_name*/errors** 파일에서 VLV 인덱스가 성공적으로 생성되었는지 확인합니다.

```
[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** 명령에서 VLV 제어를 사용하여 디렉터리에서 특정 레코드만 쿼리합니다.

```
# 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
```

이 예에서는 **ou=People,dc=example,dc=example,dc=com**의 적어도 **uid=user072** 에 지속적으로 **uid= user001** 이라는 항목이 있다고 가정합니다.

추가 리소스

- **ldapsearch(1)** 매뉴얼 페이지에 **-E** 매개변수 설명입니다.
- ldapsearch 명령의 VLV 제어

4.4. VLV 쿼리의 속도를 개선하기 위해 웹 콘솔을 사용하여 VLV 인덱스 생성

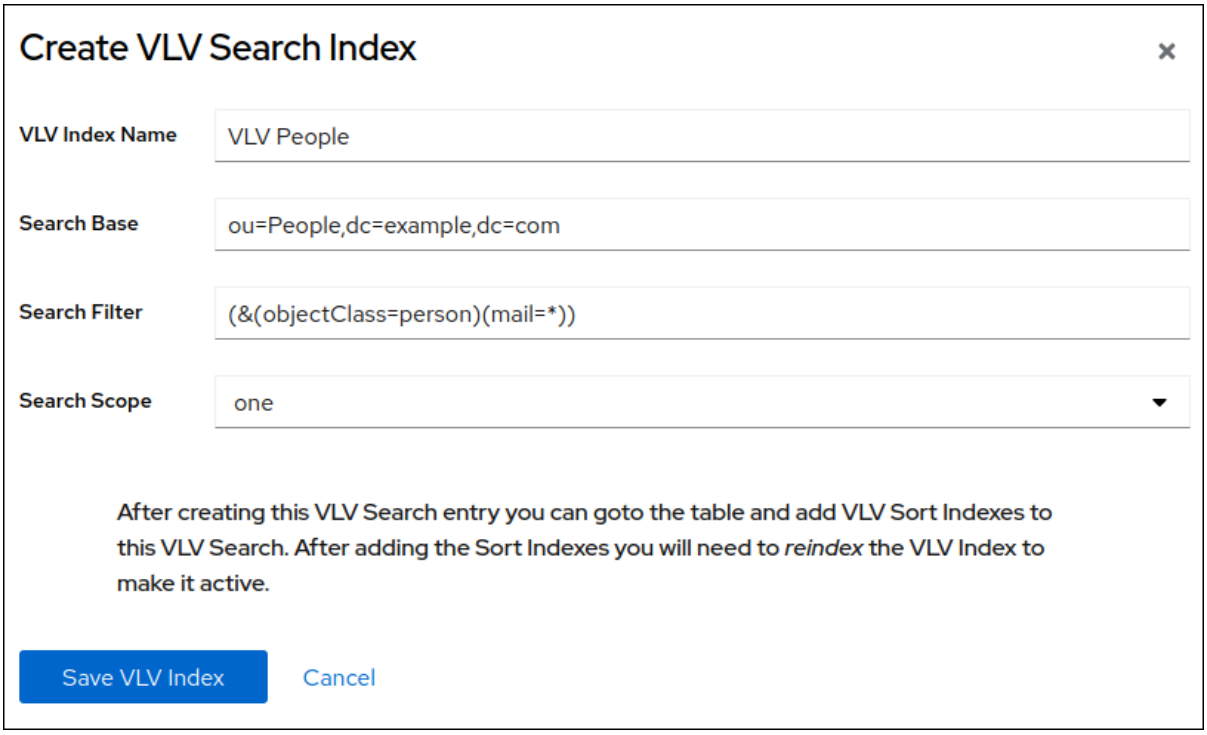
다음 절차에 따라 **mail** 속성이 포함되어 **objectClass** 속성을 포함하는 **ou=People,dc=example,dc=com** 의 항목에 대해 검색 인덱스라고도 하는 가상 목록 보기(VLV) 인덱스를 생성합니다.

사전 요구 사항

- 웹 콘솔에서 인스턴스에 로그인되어 있습니다.
- 클라이언트 애플리케이션에서 VLV 제어를 사용합니다.
- 클라이언트 애플리케이션은 대규모 검색 결과의 연속 하위 집합을 쿼리해야 합니다.
- 디렉터리에는 많은 수의 항목이 포함되어 있습니다.

절차

1. **Database** → **Suffixes** → ***dc=example,dc=com*** → **VLV Indexes** 로 이동합니다.
2. **VLV** 인덱스 생성을 클릭하고 필드를 작성합니다.



- **VLV Index Name:** 검색 항목의 이름입니다. 이것은 어떤 이름일 수 있습니다.

- 검색 기반: **VLV** 인덱스의 기본 **DN**입니다. **Directory Server**는 이 항목에 **VLV** 인덱스를 생성합니다.

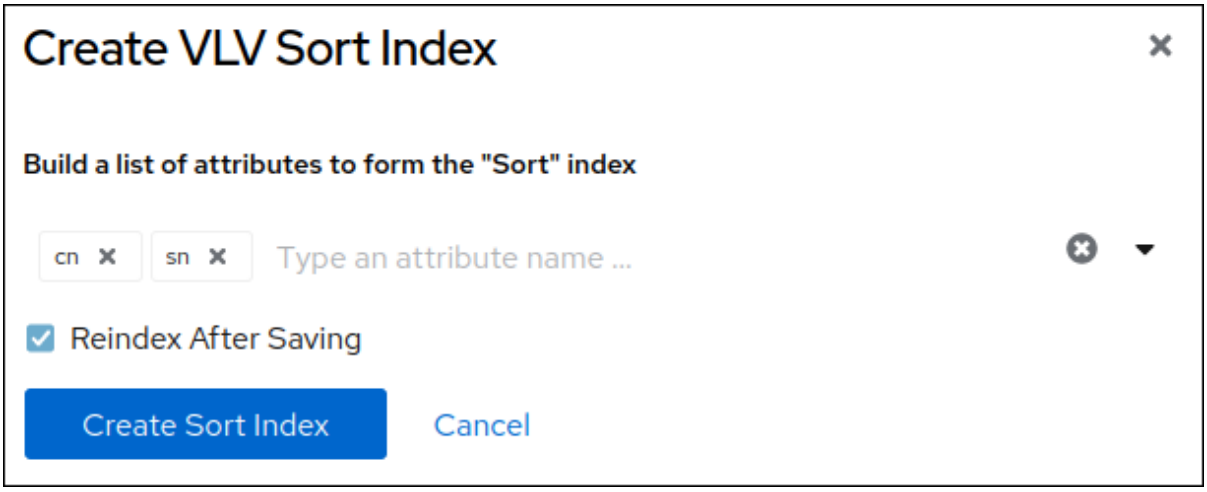
- **Search Filter:** 필터 디렉터리 서버는 **VLV** 인덱스를 생성할 때 적용됩니다. 이 필터와 일치하는 항목만 인덱스의 일부가 됩니다.

- 검색 범위: **VLV** 인덱스에서 항목을 실행할 검색 범위입니다.

3. **Save VLV Index** 를 클릭합니다.

4. 정렬 인덱스 생성을클릭합니다.

5. 속성 이름을 입력하고 저장 후 다시 인덱스 를 선택합니다.



6. 정렬 인덱스 생성을 클릭합니다.

검증

1. **Monitoring → Logging → Errors Log** 로 이동하여 VLV 인덱스가 성공적으로 생성되었는지 확인합니다.

```
[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** 명령에서 VLV 제어를 사용하여 디렉터리에서 특정 레코드만 쿼리합니다.

```
# 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
```

이 예에서는 ou=People,dc=example,dc=example,dc=com의 적어도 uid=user072 에 지속적으로 uid= user001 이라는 항목이 있다고 가정합니다.

추가 리소스

- ldapsearch(1) 매뉴얼 페이지에 -E 매개변수 설명입니다.
- ldapsearch 명령의 VLV 제어