



Hadoop Foreign Data Wrapper Guide

リリース **2.0.7**

Hadoop Foreign Data Wrapper User's Guide

2021 年 02 月 14 日

目次

第 1 章	What' s New	3
第 2 章	Requirements Overview	5
2.1	サポートされているバージョン	5
2.2	サポートされているプラットフォーム	5
第 3 章	Architecture Overview	7
第 4 章	Supported Authentication Methods	9
4.1	LDAP 認証を使用する	9
4.2	NOSASL 認証の使用	11
第 5 章	Installing the Hadoop Foreign Data Wrapper	13
5.1	RPM パッケージを使用した Hadoop 外部データラッパーのインストール	13
5.1.1	RHEL7 で	13
5.1.2	RHEL8 で	16
5.1.3	CentOS7 で	18
5.1.4	CentOS8 で	20
5.2	Debian または Ubuntu ホストへの Hadoop 外部データラッパーのインストール	22
第 6 章	Features of the Hadoop Foreign Data Wrapper	25
6.1	自動化されたクリーンアップ	25
第 7 章	Configuring the Hadoop Foreign Data Wrapper	27
7.1	エクステンションを作成	28
7.2	サーバーの作成	29
7.3	ユーザーマッピングの作成	32
7.4	外部テーブルの作成	34
7.4.1	データ型マッピング	37
第 8 章	Using the Hadoop Foreign Data Wrapper	39
8.1	Hadoop 上の ApacheHive で HDFSFDW を使用する	39
8.2	Hadoop 上の ApacheSpark で HDFSFDW を使用する	43
第 9 章	Identifying the Hadoop Foreign Data Wrapper Version	47

第 10 章 Conclusion	49
索引	51

HadoopForeignDataWrapper (`hdfs_fdw`) は、EDBPostgresAdvancedServer から Hadoop ファイルシステムにあるデータにアクセスできるようにする Postgres 拡張機能です。外部データラップにより、Hadoop ファイルシステムは読み取り専用のデータソース、Postgres の関数やユーティリティで使用したり、Postgres ホストにある他のデータと組み合わせて使用したりできます。

HadoopForeignDataWrapper は、RPM パッケージでインストールできます。EDB ウェブサイト <<https://www.enterprisedb.com/software-downloads-postgres/>> からインストーラをダウンロードできます。

このガイドでは、Postgres という用語を使用して、EDBPostgresAdvancedServer のインスタンスを指します。

第 1 章

What's New

Hadoop 外部データラッパー 2.0.7 を作成するために、次の機能が追加されました。

- EDBPostgresAdvancedServer13 のサポート。
- Ubuntu20.04LTS プラットフォームのサポート。

第 2 章

Requirements Overview

2.1 サポートされているバージョン

HadoopForeignDataWrapper は、EDBPostgresAdvancedServer9.5 以上で認定されています。

2.2 サポートされているプラットフォーム

HadoopForeignDataWrapper は、次のプラットフォームでサポートされています。

Linux x86-64

- RHEL8.x および 7.x
- CentOS8.x および 7.x
- OEL8.x および 7.x
- Ubuntu20.04 および 18.04LTS
- Debian10.x および 9.x

Linux on IBM Power8/9 (LE)

- RHEL7.x

HadoopForeignDataWrapper は、HiveServer2 インタフェースを使用する Hadoop ファイルシステムまたは Spark-ThriftServer を使用する ApacheSpark の使用をサポートしています。

第 3 章

Architecture Overview

Hadoop は、分散ファイルシステムにラージデータセットを保存できるフレームワークです。

Hadoop データラッパーは、Hadoop ファイルシステムと Postgres データベース間のインタフェースを提供します。Hadoop データラッパーは、Postgres SELECT ステートメントを、HiveQL または SparkSQL インタフェースで認識されるクエリーに変換します。

可能であれば、ForeignDataWrapper は、Hive または Spark サーバーに、SELECT ステートメントの WHERE 句に関連付けられたアクションを実行するように要求します。WHERE 句をプッシュダウンすると、ネットワークを移動するデータの量が減るため、パフォーマンスが向上します。

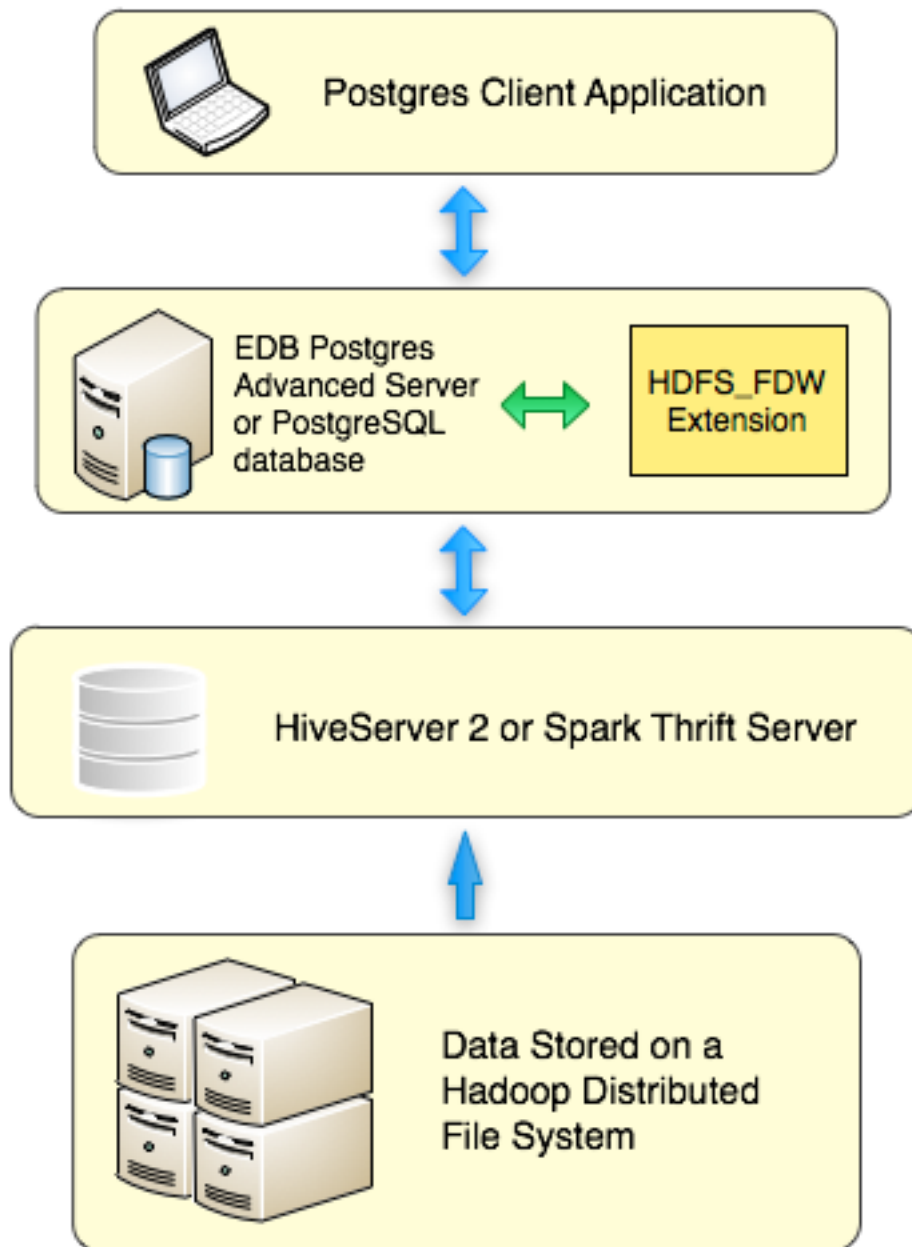


図 1 *Postgres* で *Hadoop* 分散ファイルシステムを使用する

第 4 章

Supported Authentication Methods

Hadoop 外部データラッパーは、NOSASL および LDAP 認証モードをサポートしています。NOSASL を使用するには、ユーザマッピングの作成中に OPTIONS を指定しないでください。LDAP 認証モードの場合、ユーザマッピングの作成時に OPTIONS に username および password を指定します。

4.1 LDAP 認証を使用する

LDAP 認証で Hadoop 外部データラッパーを使用する場合、最初に Hive Server または Spark Server を設定して LDAP 認証を使用する必要があります。構成されたサーバーは、LDAP サーバーの接続詳細を含む hive-site.xml ファイルを提供する必要があります。例:

```
<property>
  <name>hive.server2.authentication</name>
  <value>LDAP</value>
  <description>
    Expects one of [nosasl, none, ldap, kerberos, pam, custom].
    Client authentication types.
    NONE: no authentication check
    LDAP: LDAP/AD based authentication
    KERBEROS: Kerberos/GSSAPI authentication
    CUSTOM: Custom authentication provider
           (Use with property hive.server2.custom.authentication.class)
    PAM: Pluggable authentication module
    NOSASL: Raw transport
  </description>
</property>
<property>
  <name>hive.server2.authentication.ldap.url</name>
  <value>ldap://localhost</value>
  <description>LDAP connection URL</description>
</property>
</property>
```

(次のページに続く)

(前のページからの続き)

```
<name>hive.server2.authentication.ldap.baseDN</name>
<value>ou=People,dc=itzgeek,dc=local</value>
<description>LDAP base DN</description>
</property>
```

次に、ハイブサーバーを起動するときに、コマンドに `hive-site.xml` ファイルへのパスを含めます。例:

```
./hive --config path_to_hive-site.xml_file --service hiveServer2
```

path_to_hive-site.xml_file は、`hive-site.xml` ファイルへの完全なパスを指定します。

ユーザマッピングを作成するときは、登録済み LDAP ユーザの名前と対応するパスワードをオプションとして指定する必要があります。詳細については、[ユーザーマッピングの作成](#) を参照してください。

4.2 NOSASL 認証の使用

Hadoop 外部データラッパーで NOSASL 認証を使用する場合、Hive Server または Spark Server で認証を None に、認証メソッドを NOSASL に設定します。あなたは、コマンドラインで Hive Server をスタートした場合、コマンドで `hive.server2.authentication` 設定パラメータが含まれます。

```
hive --service hiveserver2 --hiveconf hive.server2.authentication=NOSASL
```


第 5 章

Installing the Hadoop Foreign Data Wrapper

HadoopForeignDataWrapper は、RPM パッケージでインストールできます。インストールプロセス中、インストーラはソフトウェアの前提条件を満たします。

5.1 RPM パッケージを使用した Hadoop 外部データラッパーのインストール

次のプラットフォームで RPM パッケージを使用して HadoopForeignDataWrapper をインストールできます。

- *RHEL7*
- *RHEL8*
- *CentOS7*
- *CentOS8*

5.1.1 RHEL7 で

HadoopForeignDataWrapper をインストールする前に、次の前提条件パッケージソフトをインストールし、EDB から認証情報を要求する必要があります。

epel-release パッケージをインストールします。

```
yum -y install https://dl.fedoraproject.org/pub/epel/epel-release-latest-7.noarch.  
↪rpm
```

オプション、エクストラ、HA リポジトリを有効にします。

```
subscription-manager repos --enable "rhel-*-optional-rpms" --enable "rhel-*-
↳extras-rpms" --enable "rhel-ha-for-rhel-*-server-rpms"
```

EDB リポジトリへのアクセスを許可する資格情報も必要です。資格情報のリクエストについては、次をご覧ください。

<https://info.enterprisedb.com/rs/069-ALB-339/images/Repository%20Access%2004-09-2019.pdf>

リポジトリの資格情報を受け取った後、次のことができます。

1. リポジトリ構成ファイルを作成します。
2. ユーザ名とパスワードを指定して、ファイルを変更します。
3. `edb-as<xx>-hdfs_fdw` をインストールします。

Creating a Repository Configuration File

リポジトリ構成ファイルを作成するには、スーパーユーザ特権を想定して、次のコマンドを呼び出します。

```
yum -y install https://yum.enterprisedb.com/edbrepos/edb-repo-latest.noarch.rpm
```

リポジトリ構成ファイルの名前付けは `edb.repo` です。ファイルは `/etc/yum.repos.d` にあります。

Modifying the file, providing your user name and password

`edb.repo` ファイルを作成した後、エディタの選択を使用して `enabled` パラメータの値が 1 であることを保証し、`baseurl` 仕様の `username` および `password` プレースホルダーを登録済み EDB ユーザの名前とパスワードに置き換えます。

```
[edb]
name=EnterpriseDB RPMs $releasever - $basearch
baseurl=https://<username>:<password>@yum.enterprisedb.com/edb/redhat/rhel-
↳$releasever-$basearch
enabled=1
gpgcheck=1
gpgkey=file:///etc/pki/rpm-gpg/ENTERPRISEDB-GPG-KEY
```

Installing Hadoop Foreign Data Wrapper

構成ファイルへの変更を保存した後、次のコマンドを使用して `HadoopForeignDataWrapper` をインストールします。

```
yum install edb-as<xx>-hdfs_fdw
```

ここで、「xx」はサーババージョン番号です。

システムで認識されないソースによって署名された RPM パッケージをインストールすると、`yum` は、ローカル

サーバにキーをインポートするパーミッションを求める場合があります。プロンプトが表示され、パッケージソフトが信頼できるソースからのものであることに満足したら、`y` と入力し、`Return` を押して続行します。

インストール中に、`yum` は解決できない依存関係に遭遇する場合があります。存在する場合は、手動で解決する必要がある依存関係のリストが提供されます。

5.1.2 RHEL8 で

HadoopForeignDataWrapper をインストールする前に、次の前提条件パッケージソフトをインストールし、EDB から認証情報を要求する必要があります。

epel-release パッケージをインストールします。

```
dnf -y install https://dl.fedoraproject.org/pub/epel/epel-release-latest-8.noarch.  
↪rpm
```

codeready-builder-for-rhel-8-/*-rpms リポジトリを有効にします。

```
ARCH=$( /bin/arch )  
subscription-manager repos --enable "codeready-builder-for-rhel-8-${ARCH}-rpms"
```

EDB リポジトリへのアクセスを許可する資格情報も必要です。資格情報のリクエストについては、次をご覧ください。

<https://info.enterprisedb.com/rs/069-ALB-339/images/Repository%20Access%202004-09-2019.pdf>

リポジトリの資格情報を受け取った後、次のことができます。

1. リポジトリ構成ファイルを作成します。
2. ユーザ名とパスワードを指定して、ファイルを変更します。
3. edb-as<xx>-hdfs_fdw をインストールします。

Creating a Repository Configuration File

リポジトリ構成ファイルを作成するには、スーパーユーザ特権を想定して、次のコマンドを呼び出します。

```
dnf -y https://yum.enterprisedb.com/edbrepos/edb-repo-latest.noarch.rpm
```

リポジトリ構成ファイルの名前付けは edb.repo です。ファイルは /etc/yum.repos.d にあります。

Modifying the file, providing your user name and password

edb.repo ファイルを作成した後、エディタの選択を使用して enabled パラメータの値が 1 であることを保証し、baseurl 仕様の username および password プレースホルダーを登録済み EDB ユーザの名前とパスワードに置き換えます。

```
[edb]  
name=EnterpriseDB RPMs $releasever - $basearch  
baseurl=https://<username>:<password>@yum.enterprisedb.com/edb/redhat/rhel-  
↪$releasever-$basearch  
enabled=1  
gpgcheck=1  
gpgkey=file:///etc/pki/rpm-gpg/ENTERPRISEDB-GPG-KEY
```

Installing Hadoop Foreign Data Wrapper

構成ファイルへの変更を保存した後、以下のコマンドを使用して HadoopForeignDataWrapper をインストールします。

```
dnf install edb-as<xx>-hdfs_fdw
```

システムで認識されないソースによって署名された RPM パッケージをインストールすると、yum は、ローカルサーバにキーをインポートするパーミッションを求める場合があります。プロンプトが表示され、パッケージソフトが信頼できるソースからのものであることに満足したら、y と入力し、Return を押して続行します。

インストール中に、yum は解決できない依存関係に遭遇する場合があります。存在する場合は、手動で解決する必要がある依存関係のリストが提供されます。

5.1.3 CentOS7 で

HadoopForeignDataWrapper をインストールする前に、次の前提条件パッケージソフトをインストールし、EDB から認証情報を要求する必要があります。

epel-release パッケージをインストールします。

```
yum -y install https://dl.fedoraproject.org/pub/epel/epel-release-latest-7.noarch.  
↪rpm
```

注釈: CentOS-Base.repo ファイル (/etc/yum.repos.d にある) で [extras] リポジトリ定義を有効にする必要がある場合があります。

EDB リポジトリへのアクセスを許可する資格情報も必要です。資格情報のリクエストについては、次をご覧ください。

<https://info.enterprisedb.com/rs/069-ALB-339/images/Repository%20Access%2004-09-2019.pdf>

リポジトリの資格情報を受け取った後、次のことができます。

1. リポジトリ構成ファイルを作成します。
2. ユーザ名前とパスワードを指定して、ファイルを変更します。
3. edb-as<xx>-hdfs_fdw をインストールします。

Creating a Repository Configuration File

リポジトリ構成ファイルを作成するには、スーパーユーザ特権を想定して、次のコマンドを呼び出します。

```
yum -y install https://yum.enterprisedb.com/edbrepos/edb-repo-latest.noarch.rpm
```

リポジトリ構成ファイルの名前付けは edb.repo です。ファイルは /etc/yum.repos.d にあります。

Modifying the file, providing your user name and password

edb.repo ファイルを作成した後、エディタの選択を使用して enabled パラメータの値が 1 であることを保証し、baseurl 仕様の username および password プレースホルダーを登録済み EDB ユーザの名前とパスワードに置き換えます。

```
[edb]  
name=EnterpriseDB RPMs $releasever - $basearch  
baseurl=https://<username>:<password>@yum.enterprisedb.com/edb/redhat/rhel-  
↪$releasever-$basearch  
enabled=1
```

(次のページに続く)

(前のページからの続き)

```
gpgcheck=1
gpgkey=file:///etc/pki/rpm-gpg/ENTERPRISEDB-GPG-KEY
```

Installing Hadoop Foreign Data Wrapper

構成ファイルへの変更を保存した後、次のコマンドを使用して `HadoopForeignDataWrapper` をインストールします。

```
yum install edb-as<xx>-hdfs_fdw
```

ここで、「xx」はサーババージョン番号です。

システムで認識されないソースによって署名された RPM パッケージをインストールすると、yum は、ローカルサーバにキーをインポートするパーミッションを求める場合があります。プロンプトが表示され、パッケージソフトが信頼できるソースからのものであることに満足したら、y と入力し、Return を押して続行します。

インストール中に、yum は解決できない依存関係に遭遇する場合があります。存在する場合は、手動で解決する必要がある依存関係のリストが提供されます。

5.1.4 CentOS8 で

HadoopForeignDataWrapper をインストールする前に、次の前提条件パッケージソフトをインストールし、EDB から認証情報を要求する必要があります。

epel-release パッケージをインストールします。

```
dnf -y install https://dl.fedoraproject.org/pub/epel/epel-release-latest-8.noarch.  
↪rpm
```

PowerTools リポジトリを有効にします。

```
dnf config-manager --set-enabled PowerTools
```

EDB リポジトリへのアクセスを許可する資格情報も必要です。資格情報のリクエストについては、次をご覧ください。

<https://info.enterprisedb.com/rs/069-ALB-339/images/Repository%20Access%202004-09-2019.pdf>

リポジトリの資格情報を受け取った後、次のことができます。

1. リポジトリ構成ファイルを作成します。
2. ユーザ名とパスワードを指定して、ファイルを変更します。
3. edb-as<xx>-hdfs_fdw をインストールします。

Creating a Repository Configuration File

リポジトリ構成ファイルを作成するには、スーパーユーザ特権を想定して、次のコマンドを呼び出します。

```
dnf -y install https://yum.enterprisedb.com/edbrepos/edb-repo-latest.noarch.rpm
```

リポジトリ構成ファイルの名前付けは edb.repo です。ファイルは /etc/yum.repos.d にあります。

Modifying the file, providing your user name and password

edb.repo ファイルを作成した後、エディタの選択を使用して enabled パラメータの値が 1 であることを保証し、baseurl 仕様の username および password プレースホルダーを登録済み EDB ユーザの名前とパスワードに置き換えます。

```
[edb]  
name=EnterpriseDB RPMs $releasever - $basearch  
baseurl=https://<username>:<password>@yum.enterprisedb.com/edb/redhat/rhel-  
↪$releasever-$basearch  
enabled=1  
gpgcheck=1  
gpgkey=file:///etc/pki/rpm-gpg/ENTERPRISEDB-GPG-KEY
```

Installing Hadoop Foreign Data Wrapper

構成ファイルへの変更を保存した後、次のコマンドを使用して `HadoopForeignDataWrapper` をインストールします。

```
dnf install edb-as<xx>-hdfs_fdw
```

ここで、「xx」はサーババージョン番号です。

システムで認識されないソースによって署名された RPM パッケージをインストールすると、yum は、ローカルサーバにキーをインポートするパーミッションを求める場合があります。プロンプトが表示され、パッケージソフトが信頼できるソースからのものであることに満足したら、y と入力し、Return を押して続行します。

インストール中に、yum は解決できない依存関係に遭遇する場合があります。存在する場合は、手動で解決する必要がある依存関係のリストが提供されます。

5.2 Debian または Ubuntu ホストへの Hadoop 外部データラッパーのインストール

Debian または Ubuntu ホストに HadoopForeignDataWrapper をインストールするには、EDB リポジトリへのアクセスを許可する資格情報が必要です。リポジトリの資格情報をリクエストするには、EDB ウェブサイト <https://www.enterprisedb.com/repository-access-request/> にアクセスしてください。

次の手順では、EDBapt リポジトリを使用して DEB パッケージをインストールする方法を説明します。コマンドを使用する場合、username および password を EDB が提供する資格情報に置き換えます。

1. スーパーユーザ特権を想定します。

sudo su-

2. EnterpriseDB リポジトリを構成します。

Debian9 および Ubuntu の場合:

```
sh -c 'echo "deb https://username:password@apt.enterprisedb.com/$(lsb_release_
↪-cs)-edb/ $(lsb_release -cs) main" > /etc/apt/sources.list.d/edb-$(lsb_
↪release -cs).list'
```

Debian10 の場合:

- a. EDB リポジトリをセットアップします。

```
sh -c 'echo "deb [arch=amd64] https://apt.enterprisedb.com/$(lsb_release -cs)-
↪edb/ $(lsb_release -cs) main" > /etc/apt/sources.list.d/edb-$(lsb_release -
↪cs).list'
```

- b. 次のコマンドで、username および password を EDB 資格情報に置き換えます。

```
sh -c 'echo "machine apt.enterprisedb.com login <username> password <password>
↪" > /etc/apt/auth.conf.d/edb.conf'
```

3. 安全な APT リポジトリのサポートをシステムに追加します。

apt-get install apt-transport-https

4. EDB 署名キーを追加しキー。

```
wget -q -O - https://username:password@apt.enterprisedb.com/edb-deb.gpg.key | apt-key add -
```

5. リポジトリのメタデータを更新します。

```
apt-getupdate
```

6. DEB パッケージをインストールします。

```
apt-getinstall db-as<xx>-hdfs-fdw
```

ここで、「xx」はサーババージョン番号です。

第 6 章

Features of the Hadoop Foreign Data Wrapper

HadoopForeignDataWrapper のキー機能は以下のとおりです。

6.1 自動化されたクリーンアップ

HadoopForeignDataWrapper では、`DROP EXTENSION` コマンドを使用して、1 回のオペレーションで外部テーブルをクリーンアップできます。この機能は、データ移行の場合など、外部テーブルが一時的な目的で設定されている場合に特に役立ちます。構文:

```
DROP EXTENSION hdfs_fdw CASCADE;
```

詳細については、「`DROPEXTENSION`<<https://www.postgresql.org/docs/current/sql-dropextension.html>>’_」を参照してください。

第 7 章

Configuring the Hadoop Foreign Data Wrapper

拡張機能と拡張機能を使用するデータベースオブジェクトを作成する前に、Postgres ホストを変更して、サポートライブラリの場所を指定する必要があります。

1. Postgres をインストールした後、次の場所にある `postgresql.conf` を変更します。

```
/var/lib/edb/as_version/data
```

2. コンフィギュレーション・ファイルの最後に `hdfs_fdw.jvm_path` パラメータを追加し、Java 仮想マシン (`libjvm.so`) の場所を指定する値を設定し、好みのエディタで設定ファイルを変更します。 `hdfs_fdw.classpath` の値を設定して、アダプターが使用する Java クラスファイルの場所を示します。各パス間のデリミタとしてコロンの (`:`) を使用しパス。例:

```
hdfs_fdw.classpath=  
'/usr/edb/as12/lib/HiveJdbcClient-1.0.jar: /home/edb/Projects/hadoop_fdw/hadoop/  
→share/hadoop/common/hadoop-common-2.6.4.jar:  
/home/edb/Projects/hadoop_fdw/apache-hive-1.0.1-bin/lib/hive-jdbc-1.0.1-  
→standalone.jar'
```

注釈: EDBAdvancedServer を使用しており、データベースに DATE 列がある場合、`postgresql.conf` ファイルに `edb_redwood_date = OFF` を設定する必要があります。

1. パラメータ値を設定した後、Postgres サーバーをリスタートします。AdvancedServer ホストでのサービスの制御の詳細については、次の場所にある『EDBPostgresAdvancedServer インストールガイド』を参照してください。

<https://www.enterprisedb.com/resources/product-documentation>

HadoopForeignDataWrapper を使用する前に、以下を行う必要があります。

1. `CREATEEXTENSION` コマンドを使用して、Postgres ホストで拡張機能を作成します。
2. `CREATESEVER` コマンドを使用して、Hadoop ファイルシステムへの接続を定義します。
3. `CREATEUSERMAPPING` コマンドを使用して、Postgres ロールをサーバーに関連付けるマッピングを定義します。
4. `CREATEFOREIGNTABLE` コマンドを使用して、Hadoop クラスターに存在するデータベースに対応する AdvancedServer データベースのテーブルを定義します。

7.1 エクステンションを作成

`CREATE EXTENSION` コマンドを使用して、`hdfs_fdw` 拡張機能を作成します。コマンドを実行するには、選択のあなたのクライアント（例、`psql` は）あなたはハイブまたはスパークサーバーを照会されるから、`Postgres>`に接続し、コマンドを呼び出すために使用します。

```
CREATE EXTENSION [IF NOT EXISTS] hdfs_fdw [WITH] [SCHEMA schema_name];
```

Parameters

`IF NOT EXISTS`

同じ名前の拡張子が既に存在する場合、エラーをスローする代わりに通知を発行するようにサーバーに指示するために、`IF NOT EXISTS` 句を含めます。

`schema_name`

オプションで、拡張機能のオブジェクトをインストールするスキーマの名前を指定します。

Example

次のコマンドは、`hdfs_fdw` `hadoop` 外部データラップをインストールします。

```
CREATE EXTENSION hdfs_fdw;
```

外部データラップ `CREATE EXTENSION` コマンドの使用の詳細については、以下を参照してください。

<https://www.postgresql.org/docs/current/static/sql-createextension.html>

7.2 サーバーの作成

CREATE SERVER コマンドを使用して、外部サーバへの接続を定義します。構文は次の構文です。

```
CREATE SERVER server_name FOREIGN DATA WRAPPER hdfs_fdw
    [OPTIONS (option 'value' [, ...])]
```

サーバーを定義するロールは、サーバーの所有者です。ALTER SERVER コマンドを使用して、外部サーバの所有権を再割り当てします。外部サーバを作成するには、CREATE SERVER コマンドで指定された外部データラッパーに対する USAGE 権限が必要です。

Parameters

server_name

server_name を使用して、外部サーバの名前を指定します。サーバー名前はデータベース内で一意である必要があります。

FOREIGN_DATA_WRAPPER

FOREIGN_DATA_WRAPPER 句を含めて、サーバーがクラスターに接続するときに hdfs_fdw 外部データラッパーを使用するように指定します。

OPTIONS

CREATE SERVER コマンドの OPTIONS 句を使用して、外部サーバの接続情報を指定します。以下を含めることができます。

オプション	説明
host	Hadoop クラスターのアドレスまたはホスト名。デフォルト値は 'ローカルホスト' です。
port	HiveThriftServer または SparkThriftServer のポート番号。デフォルトは '10000' です。
client_type	クライアントタイプとして hiveserver2 または spark を指定します。Spark で ANALYZE ステートメントを使用するには、spark の値を指定する必要があります。client_type の値を指定しない場合、デフォルト値は hiveserver2 です。
auth_type	クライアントの認証タイプ。LDAP または NOSASL を指定します。auth_type を指定しない場合、データラッパーはユーザマッピングに基づいて auth_type 値を決定します。 - ユーザマッピングにユーザ名前とパスワードが含まれる場合、データラッパーは LDAP 認証を使用します。 - ユーザマッピングにユーザ名前とパスワードが含まれていない場合、データラッパーは NOSASL 認証を使用します。
connect_timeout	接続の試行がタイムアウトするまでの時間の長さ。デフォルト値は「300」秒です。
fetch_size	JDBC API setFetchSize のパラメータとして提供されるユーザー指定の値。デフォルト値は「10,000」です。
log_remote_sql	真、ロギングにはリモートハイブサーバーで実行された SQL コマンドとスキャンが繰り返される回数が含まれます。デフォルトは「false」です。
query_timeout	query_timeout を使用して、Hive サーバーで要求が満たされない場合にリクエストがタイムアウトするまでの秒数を指定します。クエリタイムアウトは、HiveJDBC ドライバではサポートされていません。
use_remote_estimate	use_remote_estimate を含めて、処理コストを見積もるときにリモートサーバで EXPLAIN コマンドを使用するようにサーバーに指示します。デフォルトでは、use_remote_estimate は false であり、リモートテーブルには「1000」行があると想定されます。

Example

次のコマンドは、hdfs_fdw 外部データラップを使用して 170.11.2.148 の IP アドレスを持つホストに接続する hdfs_server 名前付けの外部サーバを作成します。

```
CREATE SERVER hdfs_server FOREIGN DATA WRAPPER hdfs_fdw OPTIONS (host '170.11.2.148',  
→port '10000', client_type 'hiveserver2', auth_type 'LDAP', connect_timeout '10000',  
→query_timeout '10000');
```

外部サーバは、Hadoop クラスター上のクライアントへの接続にデフォルトポート（10000）を使用します。接続は LDAP サーバーを使用します。

CREATE SERVER コマンドの使用の詳細については、以下を参照してください。

<https://www.postgresql.org/docs/current/static/sql-createserver.html>

7.3 ユーザーマッピングの作成

CREATE USER MAPPING コマンドを使用して、Postgres ロールを外部サーバに関連付けるマッピングを定義します。

```
CREATE USER MAPPING FOR role_name SERVER server_name
    [OPTIONS (option 'value' [, ...])];
```

そのサーバのユーザーマッピングを作成するには、外部サーバの所有者である必要があります。

ノート:HadoopForeignDataWrapper は NOSASL および LDAP 認証をサポートしています。LDAP 認証を使用するサーバのユーザーマッピングを作成する場合は、OPTIONS 句を使用して、既存の LDAP ユーザの接続資格情報（ユーザー名とパスワード）を提供します。サーバが NOSASL 認証を使用する場合は、ユーザーマッピングを作成するときに OPTIONS 句を省略します。

Parameters

role_name

role_name を使用して、外部サーバに関連付けられるロールを指定します。

server_name

server_name を使用して、Hadoop クラスターへの接続を定義するサーバの名前を指定します。

OPTIONS

OPTIONS 句を使用して、外部サーバの接続情報を指定します。LDAP 認証を使用している場合、以下を提供します。

username :LDAP サーバー上のユーザの名前。

password :ユーザー名に関連付けられたパスワード。

ユーザ名前とパスワードを指定しない場合、データラッパーは NOSASL 認証を使用します。

Example

次のコマンドは、enterprisedb 名前付けのロールのユーザーマッピングを作成します。マッピングは hdfs_server 名前付けのサーバに関連付けられます。

```
CREATE USER MAPPING FOR enterprisedb SERVER hdfs_server;
```

データベースホストが LDAP 認証を使用する場合、ユーザーマッピングの作成時に接続資格情報を提供します。

```
CREATE USER MAPPING FOR enterprisedb SERVER hdfs_server OPTIONS (username 'alice',
↪password 'lsafepwd');
```

このコマンドは、`hdfs_server` という名前付けのサーバーに関連付け名前付け `enterprisedb` という名前のロールのユーザマッピングを作成します。LDAP サーバーに接続すると、Hive または Spark サーバーは `alice` として認証し、パスワード `1safepwd` を提供します。

CREATE USER MAPPING コマンドの詳細については、以下を参照してください。

<https://www.postgresql.org/docs/current/static/sql-createusermapping.html>

7.4 外部テーブルの作成

外部テーブルは、Hadoop ホストにあるテーブルへのポインタです。Postgres サーバーで外部テーブル定義を作成する前に、Hive または Spark サーバーに接続してテーブルを作成します。テーブルの列は、Postgres サーバーのテーブルの列にマップされます。次に、CREATE FOREIGN TABLE コマンドを使用して、Hadoop ホストにあるテーブルに対応する列を持つ Postgres サーバー上のテーブルを定義します。構文は次の構文です。

```
CREATE FOREIGN TABLE [ IF NOT EXISTS ] table_name ( [
  { column_name data_type [ OPTIONS ( option 'value' [, ... ] ) ] [ COLLATE collation_
↪ ] [ column_constraint [ ... ] ]
  | table_constraint }
  [, ... ]
] )
[ INHERITS ( parent_table [, ... ] ) ]
SERVER server_name [ OPTIONS ( option 'value' [, ... ] ) ]
```

column_constraint は次のとおりです。

```
[ CONSTRAINT constraint_name ]
{ NOT NULL | NULL | CHECK (expr) [ NO INHERIT ] | DEFAULT default_expr }
```

table_constraint は次のとおりです。

```
[ CONSTRAINT constraint_name ] CHECK (expr) [ NO INHERIT ]
```

Parameters

table_name

外部テーブルの名前を指定します。スキーマ名を含めて、外部テーブルが存在するスキーマを指定します。

IF NOT EXISTS

同じ名前のテーブルが既に存在する場合、エラーをスローしないようにサーバーに指示するには、IF NOT EXISTS 句を含めます。同じ名前のテーブルが存在する場合、サーバーは通知を発行します。

column_name

新しいテーブルの列の名前を指定します。各列は、Hive または Spark サーバーで説明されている列に対応する必要があります。

data_type

列のデータタイプを指定します。可能であれば、Postgres サーバーと Hive または Spark サーバーの各列に同じデータタイプを指定します。同じ名前のデータタイプが使用できない場合、Postgres サーバーはデータ

タイプを **Hive** または **Spark** サーバーと互換性のある型にキャストしようとします。サーバーが互換性のあるデータタイプを識別できない場合、エラーを結果ます。

COLLATE collation

COLLATE 句を含めて、列に照合順序を割り当てます。指定しない場合、列データ型のデフォルトの照合順序が使用されます。

INHERITS (parent_table [, ...])

INHERITS 句を含めて、新しい外部テーブルがすべての列を自動的に継承テーブルのリストを指定します。親テーブルは、プレーンテーブルでも外部テーブルでもかまいません。

CONSTRAINT constraint_name

列または表の制約のオプショナル名前を指定します。指定しない場合、サーバーは制約名前を生成し制約。

NOT NULL

NOT NULL キーワードを含めて、列に **null** 値を含めることを許可しないことを示します。

NULL

NULL キーワードを含めて、列に **null** 値を含めることが許可されていることを示します。これがデフォルト。

CHECK (expr) [NO INHERIT]

CHECK 句を使用して、テーブルの行が満たす必要があるブール結果を生成する式を指定します。テーブル制約に現れる式はマルチプルの列をリファレンスすることができながら、列制約として指定された検査制約は、その列の値のみをリファレンスすべきです。

CHECK 式には、サブクエリを含めたり、現在の行の列以外の変数を参照したり式ことはできません。

制約が子テーブルに伝播しないように指定するには、**NO INHERIT** キーワードを含めます。

DEFAULT default_expr

DEFAULT 句を含めて、カラム定義が表示されるカラムのデフォルトのデータ値を指定します。デフォルト式のデータタイプは、列のデータタイプとマッチする必要があります。

SERVER server_name [OPTIONS (option 'value' [, ...])]

Hadoop ファイルシステムにあるテーブルをクエリーできる外部テーブルを作成するには、**SERVER** 句を含め、**Hadoop** データアダプターを使用する外部サーバの **server_name** を指定します。

OPTIONS 句を使用して、次の **options** および対応する値を指定します。

オプション	value
dbname	Hive サーバー上のデータベースの名前。データベース名前が必要です。
table_name	Hive サーバー上のテーブルの名前。デフォルトは外部テーブルの名前です。

Example

分散ファイルシステムに保存されているデータを使用するには、Hadoop テーブルの列を Postgres テーブルの列にマッピングするテーブルを Postgres ホストに作成する必要があります。例、次のように定義された Hadoop のテーブルについて:

```
CREATE TABLE weblogs (  
  client_ip          STRING,  
  full_request_date  STRING,  
  day                STRING,  
  month              STRING,  
  month_num          INT,  
  year               STRING,  
  hour               STRING,  
  minute             STRING,  
  second             STRING,  
  timezone           STRING,  
  http_verb          STRING,  
  uri                STRING,  
  http_status_code   STRING,  
  bytes_returned     STRING,  
  referrer           STRING,  
  user_agent         STRING)  
row format delimited  
fields terminated by '\t';
```

あなたは、Postgres 上の同等のテーブルを作成し、Postgres のサーバ上でコマンドを実行必要があります。

```
CREATE FOREIGN TABLE weblogs  
(  
  client_ip          TEXT,  
  full_request_date  TEXT,  
  day                TEXT,  
  Month              TEXT,  
  month_num          INTEGER,  
  year               TEXT,  
  hour               TEXT,  
  minute             TEXT,  
  second             TEXT,  
  timezone           TEXT,  
  http_verb          TEXT,  
  uri                TEXT,
```

(次のページに続く)

(前のページからの続き)

```

http_status_code      TEXT,
bytes_returned        TEXT,
referrer              TEXT,
user_agent            TEXT
)
SERVER hdfs_server
    OPTIONS (dbname 'webdata', table_name 'weblogs');

```

SERVER 句を含めて、Hadoop ファイルシステムに格納されているデータベースの名前（webdata）と、Postgres サーバーのテーブルに対応するテーブルの名前（weblogs）を指定します。

CREATE FOREIGN TABLE コマンドの使用の詳細については、以下を参照してください。

<https://www.postgresql.org/docs/current/static/sql-createforeigntable.html>

7.4.1 データ型マッピング

外部データラップを使用する場合、Hive サーバーにあるテーブルをミラーリングするテーブルを Postgres サーバーに作成する必要があります。Hadoop データラッパーは、次の Hive データ型をターゲットの Postgres 型に自動的に変換します。

Hive	Postgres
BIGINT	BIGINT/INT8
BOOLEAN	BOOL/BOOLEAN
BINARY	BYTEA
CHAR	CHAR
DATE	DATE
DOUBLE	FLOAT8
FLOAT	FLOAT/FLOAT4
INT/INTEGER	INT/INTEGER/INT4
SMALLINT	SMALLINT/INT2
STRING	TEXT
TIMESTAMP	TIMESTAMP
TINYINT	INT2
VARCHAR	VARCHAR

第 8 章

Using the Hadoop Foreign Data Wrapper

ApacheHive または ApacheSpark を介して HadoopForeignDataWrapper を使用できます。Hive と Spark はどちらも、メタデータを構成済みのメタストアに格納します。ここで、データベースとテーブルは HiveQL を使用して作成されます。

8.1 Hadoop 上の ApacheHive で HDFSFDW を使用する

Apache Hive™ データウェアハウスソフトウェアは、分散ストレージラージデータセットのクエリと管理を容易にします。Hive は、このデータに構造をプロジェクトし、HiveQL と呼ばれる SQL に似た言語を使用してデータを照会するメカニズムを提供しクエリー。同時に、この言語を使用すると、従来のマップ/reduce プログラマーは、HiveQL でこのロジックを表現するのが不便または非効率な場合に、カスタムマッパーとリデューサーをプラグインできます。

Hive には 2 つのバージョンがあります- HiveServer1 と HiveServer2 は、「ApacheHive ウェブサイト <<https://hive.apache.org/downloads.html>>」からダウンロードできます。

注釈: Hadoop 外部データラッパーは HiveServer2 のみをサポートします。

Hadoop 上で ApacheHive で HDFSFDW を使用するには:

ステップ 1:Download [weblogs_parse](#) and follow instructions from [Wiki Pentaho website](#).

ステップ 2:次のコマンドを使用して weblog_parse.txt ファイルをアップロードします。

```
hadoop fs -mkdir /weblogs
hadoop fs -mkdir /weblogs/parse
hadoop fs -put weblogs_parse.txt /weblogs/parse/part-00000
```

手順 3:次のコマンドを使用して、HiveServer を起動します（まだ実行されていない場合）。

```
$HIVE_HOME/bin/hiveserver2
```

または

```
$HIVE_HOME/bin/hive --service hiveserver2
```

ステップ 4: ハイブビーラインクライアントを使用して HiveServer2 に接続します。例:

```
$ beeline
Beeline version 1.0.1 by Apache Hive
beeline> !connect jdbc:hive2://localhost:10000/default;auth=noSasl
```

ステップ 5: Hive でテーブルを作成します。

```
CREATE TABLE weblogs (
  client_ip          STRING,
  full_request_date  STRING,
  day                STRING,
  month              STRING,
  month_num          INT,
  year               STRING,
  hour               STRING,
  minute             STRING,
  second             STRING,
  timezone           STRING,
  http_verb          STRING,
  uri                STRING,
  http_status_code   STRING,
  bytes_returned     STRING,
  referrer           STRING,
  user_agent         STRING)
row format delimited
fields terminated by '\t';
```

ステップ 6: ウェブログテーブルにデータをロードします。

```
hadoop fs -cp /weblogs/parse/part-00000 /user/hive/warehouse/weblogs/
```

ステップ 7: Postgres からデータにアクセスします。これで、Postgres の weblog テーブルを使用できます。psql を使用して接続したら、次の手順に従います。

```
-- set the GUC variables appropriately, e.g. :
hdfs_fdw.jvmpath='/home/edb/Projects/hadoop_fdw/jdk1.8.0_111/jre/lib/amd64/server/'
hdfs_fdw.classpath='/usr/local/edb/lib/postgresql/HiveJdbcClient-1.0.jar:
                  /home/edb/Projects/hadoop_fdw/hadoop/share/hadoop/common/hadoop-
↪common-2.6.4.jar:
                  /home/edb/Projects/hadoop_fdw/apache-hive-1.0.1-bin/lib/hive-jdbc-
↪1.0.1-standalone.jar'
```

(次のページに続く)

(前のページからの続き)

```
-- load extension first time after install
CREATE EXTENSION hdfs_fdw;

-- create server object
CREATE SERVER hdfs_server
    FOREIGN DATA WRAPPER hdfs_fdw
    OPTIONS (host '127.0.0.1');

-- create user mapping
CREATE USER MAPPING FOR postgres
    SERVER hdfs_server OPTIONS (username 'hive_username', password 'hive_password');

-- create foreign table
CREATE FOREIGN TABLE weblogs
(
    client_ip          TEXT,
    full_request_date  TEXT,
    day                TEXT,
    Month              TEXT,
    month_num          INTEGER,
    year               TEXT,
    hour               TEXT,
    minute             TEXT,
    second             TEXT,
    timezone           TEXT,
    http_verb          TEXT,
    uri                TEXT,
    http_status_code   TEXT,
    bytes_returned     TEXT,
    referrer           TEXT,
    user_agent         TEXT
)
SERVER hdfs_server
    OPTIONS (dbname 'default', table_name 'weblogs');

-- select from table
postgres=# SELECT DISTINCT client_ip IP, count(*)
          FROM weblogs GROUP BY IP HAVING count(*) > 5000 ORDER BY 1;
      ip          | count
-----+-----
13.53.52.13       |  5494
14.323.74.653     | 16194
322.6.648.325     | 13242
325.87.75.336     |   6500
325.87.75.36      |   6498
361.631.17.30     | 64979
```

(次のページに続く)

(前のページからの続き)

```
363.652.18.65      | 10561
683.615.622.618   | 13505
(8 rows)

-- EXPLAIN output showing WHERE clause being pushed down to remote server.
EXPLAIN (VERBOSE, COSTS OFF) SELECT client_ip, full_request_date, uri FROM weblogs_
↳WHERE http_status_code = 200;

                                QUERY PLAN

-----
↳-----
Foreign Scan on public.weblogs
  Output: client_ip, full_request_date, uri
  Remote SQL: SELECT client_ip, full_request_date, uri FROM default.weblogs WHERE_
↳((http_status_code = '200'))
(3 rows)
```

8.2 Hadoop 上の ApacheSpark で HDFSFDW を使用する

ApacheSpark™ は、さまざまなユースケースをサポートする汎用分散コンピューティングフレームワークです。リアルタイムストリーミングと、高速、使いやすさ、高度な分析機能を備えたバッチ処理を提供します。Spark は Hadoop、HBASE、Cassandra、S3 などのサードパーティストレージプロバイダーに依存しているため、ストレージレイヤーを提供しません。Spark は Hadoop とシームレスに統合し、既存のデータを処理できます。SparkSQL は、HiveQL と 100 % 互換性があり、Spark Thrift Server を使用して、Hiveserver2 の代替として使用できます。

Hadoop 上で ApacheSpark で HDFSFDW を使用するには:

手順 1: ローカルモードで ApacheSpark をダウンロードしてインストールします。

ステップ 2: フォルダー \$SPARK_HOME/conf に、次の行を含むファイル spark-defaults.conf を作成します。

```
spark.sql.warehouse.dir hdfs://localhost:9000/user/hive/warehouse
```

デフォルトでは、Spark はメタデータとデータ自分自身 (Spark のウェアハウスと呼ばれます) の両方に derby を使用します。Spark で Hadoop をウェアハウスとして使用するには、このプロパティを追加する必要があります。

ステップ 3: SparkThrift サーバーを起動します。

```
./start-thriftserver.sh
```

ステップ 4: SparkThrift サーバーが実行され、ログファイルに書き込まれていることを確認します。

ステップ 5: 次のエントリを含むローカルファイル (names.txt) を作成します。

```
$ cat /tmp/names.txt
1,abcd
2,pqrs
3,wxyz
4,a_b_c
5,p_q_r
,
```

ステップ 6: Spark beeline クライアントを使用して SparkThriftServer2 に接続します。例:

```
$ beeline
Beeline version 1.2.1.spark2 by Apache Hive
beeline> !connect jdbc:hive2://localhost:10000/default;auth=noSasl org.apache.hive.
↪ jdbc.HiveDriver
```

ステップ 7: Spark でサンプルデータを準備します。beeline コマンドラインツールで次のコマンドを実行します。

```
./beeline
Beeline version 1.2.1.spark2 by Apache Hive
beeline> !connect jdbc:hive2://localhost:10000/default;auth=noSasl org.apache.hive.
↪ jdbc.HiveDriver
Connecting to jdbc:hive2://localhost:10000/default;auth=noSasl
Enter password for jdbc:hive2://localhost:10000/default;auth=noSasl:
Connected to: Spark SQL (version 2.1.1)
Driver: Hive JDBC (version 1.2.1.spark2)
Transaction isolation: TRANSACTION_REPEATABLE_READ
0: jdbc:hive2://localhost:10000> create database my_test_db;
+-----+---+
| Result |
+-----+---+
+-----+---+
No rows selected (0.379 seconds)
0: jdbc:hive2://localhost:10000> use my_test_db;
+-----+---+
| Result |
+-----+---+
+-----+---+
No rows selected (0.03 seconds)
0: jdbc:hive2://localhost:10000> create table my_names_tab(a int, name string)
                                row format delimited fields terminated by ' ';
+-----+---+
| Result |
+-----+---+
+-----+---+
No rows selected (0.11 seconds)
0: jdbc:hive2://localhost:10000>

0: jdbc:hive2://localhost:10000> load data local inpath '/tmp/names.txt'
                                into table my_names_tab;
+-----+---+
| Result |
+-----+---+
+-----+---+
No rows selected (0.33 seconds)
0: jdbc:hive2://localhost:10000> select * from my_names_tab;
+-----+-----+---+
| a      | name  |
+-----+-----+---+
| 1      | abcd  |
| 2      | pqrs  |
| 3      | wxyz  |
| 4      | a_b_c |
| 5      | p_q_r |
| NULL   | NULL  |
+-----+-----+---+
```

次のコマンドは、Hadoop の対応するファイルをリストします。

```
$ hadoop fs -ls /user/hive/warehouse/
Found 1 items
drwxrwxrwx   - org.apache.hive.jdbc.HiveDriver supergroup 0 2020-06-12 17:03 /user/
↳hive/warehouse/my_test_db.db

$ hadoop fs -ls /user/hive/warehouse/my_test_db.db/
Found 1 items
drwxrwxrwx   - org.apache.hive.jdbc.HiveDriver supergroup 0 2020-06-12 17:03 /user/
↳hive/warehouse/my_test_db.db/my_names_tab
```

ステップ 8:Postgres からデータにアクセスします。psql を使用して Postgres に接続します。

```
-- set the GUC variables appropriately, e.g. :
hdfs_fdw.jvmpath='/home/edb/Projects/hadoop_fdw/jdk1.8.0_111/jre/lib/amd64/server/'
hdfs_fdw.classpath='/usr/local/edb/lib/postgresql/HiveJdbcClient-1.0.jar:
                    /home/edb/Projects/hadoop_fdw/hadoop/share/hadoop/common/hadoop-
↳common-2.6.4.jar:
                    /home/edb/Projects/hadoop_fdw/apache-hive-1.0.1-bin/lib/hive-jdbc-
↳1.0.1-standalone.jar'

-- load extension first time after install
CREATE EXTENSION hdfs_fdw;

-- create server object
CREATE SERVER hdfs_server
    FOREIGN DATA WRAPPER hdfs_fdw
    OPTIONS (host '127.0.0.1', port '10000', client_type 'spark', auth_type 'NOSASL');

-- create user mapping
CREATE USER MAPPING FOR postgres
    SERVER hdfs_server OPTIONS (username 'spark_username', password 'spark_password');

-- create foreign table
CREATE FOREIGN TABLE f_names_tab( a int, name varchar(255)) SERVER hdfs_svr
    OPTIONS (dbname 'testdb', table_name 'my_names_tab');

-- select the data from foreign server
select * from f_names_tab;
 a | name
---+-----
 1 | abcd
 2 | pqrs
 3 | wxyz
 4 | a_b_c
 5 | p_q_r
 0 |
(6 rows)
```

(次のページに続く)

(前のページからの続き)

```
-- EXPLAIN output showing WHERE clause being pushed down to remote server.
EXPLAIN (verbose, costs off) SELECT name FROM f_names_tab WHERE a > 3;

               QUERY PLAN

-----
Foreign Scan on public.f_names_tab
  Output: name
  Remote SQL: SELECT name FROM my_test_db.my_names_tab WHERE ((a > '3'))
(3 rows)
```

注釈: SparkThriftServer は HiveThriftServer と互換性があるため、外部サーバの作成中に同じポートが使用されていました。Hiveserver2 を使用するアプリケーションは、ANALYZE コマンドの動作と NOSASL の場合の接続文字列を除き、Spark で動作します。Hive を Spark に置き換える場合は、ALTER SERVER を使用し、client_type オプションを変更することをお勧めします。

第 9 章

Identifying the Hadoop Foreign Data Wrapper Version

Hadoop の外国データラッパーを使用すると、データラッパーの .so ファイルの現在インストールされているバージョンを識別するために使用できるファンクションが含まれています。ファンクションを使用するには、Postgres サーバーに接続し、次のように入力します。

```
SELECT hdfs_fdw_version();
```

ファンクションはバージョン番号を返します。

```
hdfs_fdw_version  
-----  
<xxxxxx>
```


第 10 章

Conclusion

Hadoop 外部データラッパーガイド

Copyright©2017-2020EnterpriseDBCorporation。

禁無断転載

EnterpriseDB

34CrosbyDrive、Suite201、Bedford、MA01730、アメリカ

T+17813573390F+19784671307E

info@enterprisedb.com

www.enterprisedb.com

- EnterpriseDB および PostgresEnterpriseManager は、EnterpriseDBCorporation の登録商標です。EDB および EDBPostgres は、EnterpriseDBCorporation の商標です。オラクルは、オラクル Inc の登録商標です。その他の商標は、それぞれの所有者の商標です。
- EDB は、存在する場合、EDBPostgres 製品のログオン UI の設計、コーディングのベストプラクティスの確立、レビュー、および入力バリデーションを行います。EDB は追加の入力コンポーネントに対して同じアプローチに従いますが、製品の性質上、限定されたバリデーションが可能な信頼できるユーザーによって入力および送信されるフリーフォーム SQL、WMI またはその他の文字列を受け入れる必要がある場合があります。このような場合、ユーザーが誤った入力や危険な入力を禁止することはできません。
- EDB は、フリーフォーム SQL または将来的に認証され、信頼されたユーザーからの他の潜在的に危険な入力を受け入れるの製品に機能を追加する権利を留保しますが、彼らは最低限のリスクの可能性を提供確保する保証に設計され、テストされているようなすべての機能を保証し、どこ可能であれば、スーパーユーザまたは同等の特権が必要です。
- EDB は、すべての潜在的な脅威を予測できることを保証しません。したがって、すべての潜在的な脆弱性が対処または検討されたことをプロセスが完全に保証することはできません。

索引

Architecture Overview, 7

centos7, 18
centos8, 20
Conclusion, 49
Configuring the Hadoop Foreign Data Wrapper, 27

Features of the Hadoop Foreign Data Wrapper, 25

Identifying the Hadoop Foreign Data Wrapper
Version, 47
Installing Hadoop Data Adapter on Debian, 22
Installing Hadoop Data Adapter on Ubuntu, 22
Installing the Hadoop Foreign Data Wrapper, 13

Requirements Overview, 5
rhel8, 16

Supported Authentication Methods, 9
supported platforms, 5
supported server versions, 5

Using the Hadoop Foreign Data Wrapper, 39

What' s New, 3