



MySQL Foreign Data Wrapper Guide

リリース 2.5.5

MySQL 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	Installing the MySQL Foreign Data Wrapper	9
4.1	RPM パッケージを使用した MySQLForeignDataWrapper のインストール	9
4.1.1	RHEL7 で	9
4.1.2	RHEL8 で	12
4.1.3	CentOS7 で	14
4.1.4	CentOS8 で	16
4.2	MySQL 外部データラッパーを Debian または Ubuntu ホストにインストールする	18
第 5 章	Features of the MySQL Foreign Data Wrapper	21
5.1	Writable FDW	21
5.2	Connection Pooling	21
5.3	WHERE Clause Push-down	22
5.4	Column Push-down	22
5.5	Prepared Statement	22
5.6	Import Foreign Schema	22
5.7	Automated Cleanup	22
第 6 章	Configuring the MySQL Foreign Data Wrapper	23
6.1	CREATE EXTENSION	23
6.2	CREATE SERVER	25
6.3	CREATE USER MAPPING	27
6.4	CREATE FOREIGN TABLE	28
6.4.1	Data Type Mappings	30
6.5	IMPORT FOREIGN SCHEMA	32
第 7 章	Example:Using the MySQL Foreign Data Wrapper	35

第 8 章	Example:Import Foreign Schema	37
第 9 章	Identifying the MySQL Foreign Data Wrapper Version	39
第 10 章	Troubleshooting	41
第 11 章	Conclusion	43
索引		45

MySQLForeignDataWrapper (`mysql_fdw`) は、MySQL データベースにあるデータに EDBPostgresAdvancedServer からアクセスできるようにする Postgres 拡張機能です。これは書き込み可能な外部データラップであり、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。

MySQLForeignDataWrapper は RPM パッケージでインストールできます。⁴EDB 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

MySQLForeignDataWrapper 2.5.5 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

MySQLForeignDataWrapper 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

MySQLForeignDataWrapper 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/7.x
- CentOS8.x/7.x
- OEL8.x/7.x
- Ubuntu20.04/18.04LTS
- Debian10.x/9.x

Linux on IBM Power8/9 (LE)

- RHEL7.x

第 3 章

Architecture Overview

MySQL データラッパーは、MySQL サーバーと Postgres データベース間のインタフェースを提供します。Postgres ステートメント（SELECT / INSERT / DELETE / UPDATE）を MySQL データベースで認識されるクエリーに変換します。

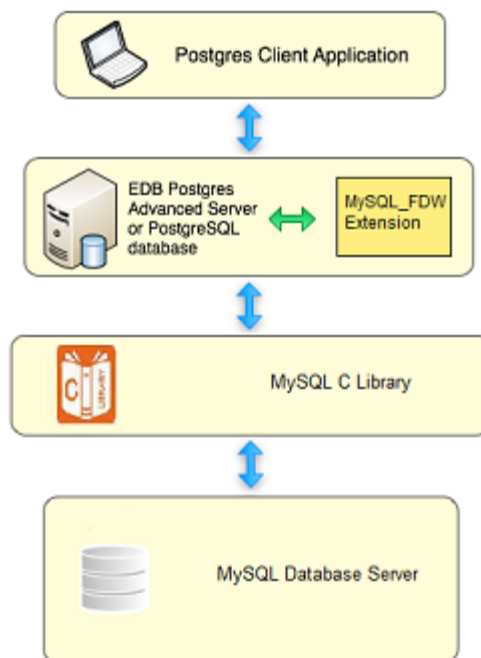


図 1 Postgres で `mysql_fdw` を使用する

第 4 章

Installing the MySQL Foreign Data Wrapper

MySQLForeignDataWrapper 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。

4.1 RPM パッケージを使用した MySQLForeignDataWrapper 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 MySQLForeignDataWrapper をインストールできます。

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

4.1.1 RHEL7 で

MySQLForeignDataWrapper 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 から認証情報をリクエスト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:

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. MySQLForeignDataWrapper 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 to provide 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 the MySQL 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 MySQLForeignDataWrapper をインストールします。

```
yum install edb-as<xx>-mysql<x>_fdw
```

ここで、「xx」はサーババージョン番号、「x」はサポートされている MySQL のリリースバージョン番号です。例、RHEL7 に MySQL5.0 をインストールします。

```
yum install edb-as<xx>-mysql5_fdw
```

注釈: MySQL8.0 および MySQL5.0RPM は、RHEL7 プラットフォームで使用できます。

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。

4.1.2 RHEL8 で

MySQLForeignDataWrapper 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 から認証情報をリクエスト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%2004-09-2019.pdf>

リポジトリの資格情報を受け取った後:

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. MySQLForeignDataWrapper 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 to provide 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 the MySQL 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 `MySQLForeignDataWrapper` をインストールします。

```
dnf install edb-as<xx>-mysql8_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。

4.1.3 CentOS7 で

MySQLForeignDataWrapper 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 から認証情報をリクエスト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:

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. MySQLForeignDataWrapper 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 to provide 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 MySQL 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 `MySQLForeignDataWrapper` をインストールします。

```
yum install edb-as<xx>-mysql<x>_fdw
```

ここで、「xx」はサーババージョン番号、「x」はサポートされている MySQL のリリースバージョン番号です。例、CentOS の 7 に MySQL5.0 をインストールします。

```
yum install edb-as<xx>-mysql5_fdw
```

注釈: MySQL8.0 および MySQL5.0RPM は、CentOS7 プラットフォームで使用できます。

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。

4.1.4 CentOS8 で

MySQLForeignDataWrapper 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 から認証情報をリクエスト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:

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. MySQLForeignDataWrapper 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 to provide 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 MySQL 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 `MySQLForeignDataWrapper` をインストールします。

```
dnf install edb-as<xx>-mysql8_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。

4.2 MySQL 外部データラッパーを Debian または Ubuntu ホストにインストールする

MySQL 外部データラッパーを Debian または Ubuntu ホストに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 Debian 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 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-get update
```

6. Debian パッケージをインストールします。

```
apt-get install edb-as<xx>-mysql-fdw
```

ここで、「xx」はサーババージョン番号です。

第 5 章

Features of the MySQL Foreign Data Wrapper

MySQLForeignDataWrapper のキー機能は次のとおりです。

5.1 Writable FDW

MySQLForeignDataWrapper を使用すると、PostgreSQL サーバーから MySQL サーバーに保存されているコンテンツを変更できます。ユーザーは、外部テーブルのデータをローカルで挿入、更新、削除する挿入により、リモート MySQL テーブルのデータを INSERT、UPDATE および DELETE できます。MySQL 外部データラッパは、Postgres 型キャストメカニズムを使用して、MySQL と Postgres データ型の間で反対の型キャストを提供します。

注釈: MySQL テーブルの最初の列には、DML が機能するための一意のプライマリキーが必要です。

こちらをご覧ください:

例:MySQL 外部データラッパの使用

Data Type Mappings

5.2 Connection Pooling

MySQL_FDW は、外部サーバーに関連付けられた外部テーブルを使用する最初のクエリー中に外部サーバへの接続を確立し外部サーバ。この接続は保持され、同じセッションの後続のクエリで再利用されます。

5.3 WHERE Clause Push-down

MySQLForeignDataWrapper では、WHERE 句を実行のために外部サーバにプッシュダウンできます。この機能は、リモートクエリを最適化して、外部サーバーから転送される行の数を減らします。

5.4 Column Push-down

MySQLForeignDataWrapper は、選択的 Column Push-down をサポートしています。その結果、クエリーは select ターゲットリストのパートである列のみを戻します。

5.5 Prepared Statement

MySQLForeignDataWrapper は準備されたステートメントをサポートします。選択クエリは、シンプルクエリープロトコルの代わりに準備されたステートメントを使用します。

5.6 Import Foreign Schema

MySQLForeignDataWrapper では、外部スキーマをインポートできます。これにより、ローカルホストはテーブル定義を MySQL サーバーから AdvancedServer ホストにインポートできます。新しい外部テーブルは、対応する列タイプと既存のローカルスキーマのリモートテーブルと同じテーブル名前で作成されます。

こちらをご覧ください:

例:外部スキーマのインポート

5.7 Automated Cleanup

MySQLForeignDataWrapperr 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、データ移行の場合のように、一時的な目的で外部テーブルが作成される場合に特に役立ちます。構文は次の構文です。

```
DROP EXTENSION mysql_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 6 章

Configuring the MySQL Foreign Data Wrapper

MySQLForeignDataWrapper 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 MySQLForeignDataWrapper 拡張機能を作成します。
2. `CREATESERVER` コマンドを使用して、MySQL サーバー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ローカルスキーマにマルチプルのリモートテーブルをインポートします。

6.1 CREATE EXTENSION

`CREATE EXTENSION` コマンドを使用して、`mysql_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）MySQL サーバを照会されるから、`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] mysql_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、MySQL 外部データラップをインストールします。

```
CREATE EXTENSION mysql_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>.

6.2 CREATE SERVER

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 mysql_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 mysql_fdw 外部データラッパ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	MySQL サーバーのアドレスまたはホスト名。デフォルト値は '127.0.0.1' です。
port	MySQL サーバーのポート番号。デフォルトは '3306' です。
secure_auth	セキュア認証を有効または無効にするために使用します。デフォルト値は '真' です。
init_command	MySQL サーバーに接続するときに実行 SQL ステートメント。
ssl_key	クライアントプライベートキーファイルのパス名前。
ssl_cert	クライアント公開キー証明書ファイルのパス名前。
ssl_ca	認証局 (CA) 証明書ファイルのパス名前。このオプションを使用する場合、サーバーが使用する証明書と同じ証明書を指定する必要があります。
ssl_capath	信頼できる SSLCA 証明書ファイルを含むディレクトリのパス名前。
ssl_cipher	SSL 暗号化に許可される暗号のリスト。
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 です。

Example

次のコマンドは、mysql_fdw 外部データラップアを使用して 127.0.0.1 の IP アドレスを持つホストに接続する mysql_server 名前付けの外部サーバを作成します。

```
CREATE SERVER mysql_server FOREIGN DATA WRAPPER mysql_fdw OPTIONS (host '127.0.0.1',  
↪port '3306');
```

外部サーバは、MySQL 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（3306）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>

6.3 CREATE USER MAPPING

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' [, ...])];
```

そのサーバーのユーザ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。

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、MySQL クラスターへの接続を定義するサーバー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。

`username`:MySQL サーバー上のユーザの名前。

`password`:ユーザ名に関連付けられたパスワード。

Example

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

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

データベースホストが安全な認証を使用する場合、ユーザマッピングを作成するときに接続資格情報を提供します。

```
CREATE USER MAPPING FOR public SERVER mysql_server OPTIONS (username 'foo', password
↪ 'bar');
```

このコマンドは、mysql_server という名前付けのサーバーに関連付け名前付け public という名前のロールのユーザマッピングを作成します。MySQL サーバーに接続すると、サーバーは foo として認証し、パスワード bar 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>

6.4 CREATE FOREIGN TABLE

外部テーブルは、MySQL 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、MySQL 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appableです。次に、CREATE FOREIGN TABLE コマンドを使用して、MySQL 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、MySQL 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。可能な場合は、Postgres サーバーと MySQL 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利用できない場合、Postgres サーバーはデータタイプを

MySQL 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' [, ... ] ) ]`

MySQL 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、MySQL 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	MySQL 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	MySQL 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。
max_blob_size	切り捨てなしで読み取る最大 BLOB サイズ。

Example

MySQL サーバーに保存されているデータを使用するには、MySQL 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 MySQL のテーブルのために:

```
CREATE TABLE warehouse (  
  warehouse_id      INT PRIMARY KEY,  
  warehouse_name    TEXT,  
  warehouse_created TIMESTAMP);
```

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 warehouse  
(  
  warehouse_id      INT,  
  warehouse_name    TEXT,  
  warehouse_created TIMESTAMP  
)  
SERVER mysql_server  
  OPTIONS (dbname 'db', table_name 'warehouse');
```

MySQL サーバとテーブルの名前 (Postgres のサーバ上のテーブルに対応することに格納されたデータベースの名前を指定する SERVER 句を含みます。

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>

注釈: MySQL 外部データラップは、書き込み機能機能をサポートします。

6.4.1 Data Type Mappings

外部データラップを使用する場合、MySQL 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。MySQL データラッパーは、次の MySQL データ型をターゲット Postgres 型に自動的に変換します。

MySQL	Postgres
BIGINT	BIGINT/INT8
BOOLEAN	SMALLINT
BLOB	BYTEA
CHAR	CHAR
DATE	DATE
DOUBLE	DOUBLEPRECISION/FLOAT8
FLOAT	FLOAT/FLOAT4
INT/INTEGER	INT/INTEGER/INT4
LONGTEXT	TEXT
SMALLINT	SMALLINT/INT2
TIMESTAMP	TIMESTAMP
VARCHAR ()	VARCHAR () /CHARCTERVARYING ()

注釈: ENUM データタイプ:

MySQL は、文字列フォームの enum 値を受け入れます。MySQL サーバーに存在する値とまったく同じ enum リストを AdvancedServer に作成する必要があります。ローカルサーバで不明な値を持つ行をフェッチしているときに、何らかの不整合があるとエラーが発生します。

また、指定された enum 値が MySQL 側に存在せず、Postgres/AdvancedServer 側に存在する場合、enum カラムの空の文字列 (' ') が MySQL 側の値として挿入されます。 ' ' などの列挙値を持つそのようなテーブルから選択するには、すべての有効な値と ' ' を使用して、Postgres 側で enum 型を作成します。

6.5 IMPORT FOREIGN SCHEMA

IMPORT FOREIGN SCHEMA コマンドを使用して、MySQL サーバーから Postgres サーバー上のテーブル定義をインポートします。新しい外部テーブルは、既存のローカスキーマのリモートテーブルと同じ列定義で作成されます。構文は次の構文です。

```
IMPORT FOREIGN SCHEMA remote_schema
  [ { LIMIT TO | EXCEPT } ( table_name [, ...] ) ]
FROM SERVER server_name
INTO local_schema
[ OPTIONS ( option 'value' [, ...] ) ]
```

Parameters

remote_schema

インポートのリモートスキーマ（MySQL データベース）を指定します。

LIMIT TO (table_name [, ...])

デフォルトでは、MySQL ホスト上の特定のデータベースに存在するすべてのビューとテーブルがインポートされます。このオプションを使用すると、テーブルのリストを指定されたサブセットに制限できます。

EXCEPT (table_name [, ...])

デフォルトでは、MySQL ホスト上の特定のデータベースに存在するすべてのビューとテーブルがインポートされます。このオプションを使用すると、指定した外部テーブルをインポートから除外できます。

SERVER server_name

外部テーブルをインポートサーバーの名前を指定します。

local_schema

インポートされた外部テーブルを作成する必要があるローカスキーマの名前を指定します。

OPTIONS

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

Option	Description
import_default	外部サーバからインポートされた外部テーブルの定義に列の 'DEFAULT' 式を含めるかどうかを制御します。デフォルトは「false」です。
import_not_null	外部サーバからインポートされた外部テーブルの定義に列「NOTNULL」コンストレインを含めるかどうかを制御します。デフォルトは「真」です。

Example

次の定義で edb データベースに作成された MySQL テーブルの場合:

```
CREATE TABLE color(cid INT PRIMARY KEY, cname TEXT);
INSERT INTO color VALUES (1, 'Red');
INSERT INTO color VALUES (2, 'Green');
INSERT INTO color VALUES (3, 'Orange');

CREATE TABLE fruit(fid INT PRIMARY KEY, fname TEXT);
INSERT INTO fruit VALUES (1, 'Orange');
INSERT INTO fruit VALUES (2, 'Mango');
```

Postgres のサーバー上の輸入に匹敵テーブルというあなたは、Postgres 上でコマンドを実行必要があります。

```
IMPORT FOREIGN SCHEMA edb FROM SERVER mysql_server INTO public;

SELECT * FROM color;

cid | cname
-----+-----
  1 | Red
  2 | Green
  3 | Orange
(3 rows)

SELECT * FROM fruit;

fid | fname
-----+-----
  1 | Orange
  2 | Mango
(2 rows)
```

このコマンドは、サーバー mysql_server 上のリモートスキーマ edb からテーブル定義をインポートし、ローカルスキーマ public に外部テーブルを作成します。

IMPORT FOREIGN SCHEMA 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-importforeignschema.html>

第 7 章

Example: Using the MySQL Foreign Data Wrapper

AdvancedServer からデータにアクセスし、psql に接続します。psql に接続したら、次の手順に従います。

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

-- create server object
CREATE SERVER mysql_server
    FOREIGN DATA WRAPPER mysql_fdw
    OPTIONS (host '127.0.0.1', port '3306');

-- create user mapping
CREATE USER MAPPING FOR postgres
    SERVER mysql_server OPTIONS (username 'foo', password 'bar');

-- create foreign table
CREATE FOREIGN TABLE warehouse
(
    warehouse_id      INT,
    warehouse_name    TEXT,
    warehouse_created TIMESTAMPTZ
)
SERVER mysql_server
    OPTIONS (dbname 'db', table_name 'warehouse');

-- insert new rows in table
INSERT INTO warehouse values (1, 'UPS', current_date);
INSERT INTO warehouse values (2, 'TV', current_date);
INSERT INTO warehouse values (3, 'Table', current_date);

-- select from table
SELECT * FROM warehouse ORDER BY 1;
```

(次のページに続く)

(前のページからの続き)

```
warehouse_id | warehouse_name | warehouse_created
-----+-----+-----
          1 | UPS             | 10-JUL-20 00:00:00
          2 | TV              | 10-JUL-20 00:00:00
          3 | Table           | 10-JUL-20 00:00:00
```

```
-- delete row from table
```

```
DELETE FROM warehouse where warehouse_id = 3;
```

```
-- update a row of table
```

```
UPDATE warehouse set warehouse_name = 'UPS_NEW' where warehouse_id = 1;
```

```
-- explain a table with verbose option
```

```
EXPLAIN VERBOSE SELECT warehouse_id, warehouse_name FROM warehouse WHERE warehouse_
↪name LIKE 'TV' limit 1;
```

QUERY PLAN

```
↪-----
Limit   (cost=10.00..11.00 rows=1 width=36)
  Output: warehouse_id, warehouse_name
    -> Foreign Scan on public.warehouse   (cost=10.00..1010.00 rows=1000 width=36)
      Output: warehouse_id, warehouse_name
      Local server startup cost: 10
      Remote query: SELECT `warehouse_id`, `warehouse_name` FROM `db`.
↪`warehouse` WHERE ((`warehouse_name` LIKE BINARY 'TV'))
```

第 8 章

Example: Import Foreign Schema

AdvancedServer からデータにアクセスし、psql に接続します。psql に接続したら、次の手順に従います。

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

-- create server object
CREATE SERVER mysql_server
    FOREIGN DATA WRAPPER mysql_fdw
    OPTIONS (host '127.0.0.1', port '3306');

-- create user mapping
CREATE USER MAPPING FOR postgres
    SERVER mysql_server OPTIONS (username 'foo', password 'bar');

-- import foreign schema
IMPORT FOREIGN SCHEMA edb FROM SERVER mysql_server INTO public;

SELECT * FROM color;
cid | cname
-----+-----
  1 | Red
  2 | Green
  3 | Orange

SELECT * FROM fruit;
fid | fname
-----+-----
  1 | Orange
  2 | Mango
```


第 9 章

Identifying the MySQL Foreign Data Wrapper Version

MySQL の外国の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 mysql_fdw_version();
```

ファンクションはバージョン番号を返します。

```
mysql_fdw_version  
-----  
<xxxxxx>
```


第 10 章

Troubleshooting

MySQL8 および MySQL_FDW の使用で問題が発生した場合、頻繁に見られる問題のソリューションのリストを以下に示します。

Authentication plugin 'caching_sha2_password' Error

```
ERROR:  failed to connect to MySQL: Authentication plugin 'caching_sha2_password'
↳ cannot be loaded: /usr/lib64/mysql/plugin/caching_sha2_password.so: cannot open
↳ shared object file: No such file or directory
```

認証プラグインを mysql_native_password として指定し、クリアテキストパスワード値を設定します。構文:

```
ALTER USER 'username'@'host' IDENTIFIED WITH mysql_native_password BY
'<password>';
```

注釈: 上記のエラーの詳細については、**MySQL8** の文書<<https://dev.mysql.com/doc/refman/8.0/en/upgrading-from-previous-series.html>>を参照してください。

第 11 章

Conclusion

MySQL 外部データラッパーガイド

Copyright©2020-2020EnterpriseDBCorporation。

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, 14

centos8, 16

Conclusion, 43

Configuring the MySQL Foreign Data Wrapper, 23

Example: Import Foreign Schema, 37

Example: Using the MySQL Foreign Data Wrapper, 35

Features of the MySQL Foreign Data Wrapper, 21

Identifying the MySQL Foreign Data Wrapper
Version, 39

Installing MySQL Data Adapter on Debian, 18

Installing MySQL Data Adapter on Ubuntu, 18

Installing the MySQL Foreign Data Wrapper, 9

Requirements Overview, 5

rhel8, 12

supported platforms, 5

supported server versions, 5

Troubleshooting, 41

What' s New, 3