



EDB

MongoDB Foreign Data Wrapper Guide

リリース **5.2.8**

EnterpriseDB

2021 年 02 月 13 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 MongoDB Foreign Data Wrapper	9
4.1	RPM パッケージを使用した MongoDB 外部データラッパーのインストール	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	Debian または Ubuntu ホストへの MongoDB 外部データラッパーのインストール	18
第 5 章	Features of the MongoDB Foreign Data Wrapper	21
5.1	書き込み可能な FDW	21
5.2	接続プーリング	21
5.3	自動化されたクリーンアップ	21
第 6 章	Configuring the MongoDB Foreign Data Wrapper	23
6.1	エクステンションを作成	23
6.2	サーバーの作成	25
6.3	ユーザーマッピングの作成	27
6.4	外部テーブルの作成	28
6.4.1	データ型マッピング	31
第 7 章	Example:Using the MongoDB Foreign Data Wrapper	33
第 8 章	Identifying the MongoDB Foreign Data Wrapper Version	37
第 9 章	Conclusion	39
	索引	41

MongoDBForeignDataWrapper (mongo_fdw) は、EDBPostgresAdvancedServer から MongoDB データベース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。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。

MongoDBForeignDataWrapper は、RPM パッケージでインストールできます。¹EDB 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

MongoDB 外部データラッパー 5.2.8 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

MongoDBForeignDataWrapper 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

MongoDB 外部データラッパーは、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

MongoDB データラッパーは、MongoDB 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 MongoDB データベースが理解できるクエリーに変換します。

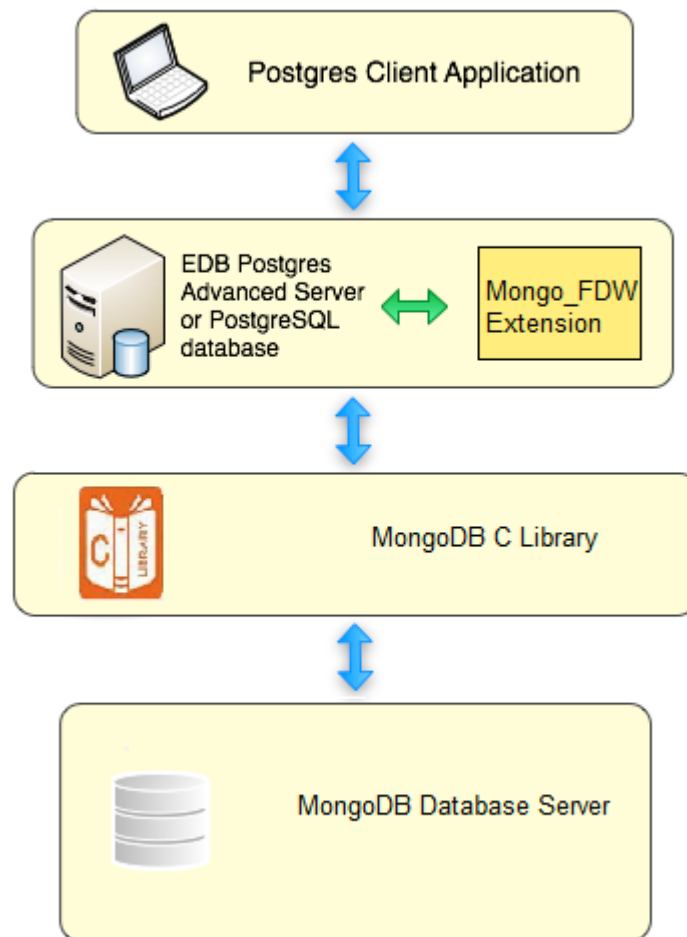


図 1 *Postgres* で *MongoDBFDW* を使用する

第 4 章

Installing the MongoDB Foreign Data Wrapper

MongoDBForeignDataWrapper 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 MongoDB 外部データラッパーのインストール

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

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

4.1.1 RHEL7 で

MongoDBForeignDataWrapper 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:

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>-mongo_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 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 MongoDB 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 `MongoDBForeignDataWrapper` をインストールします。

```
yum install edb-as<xx>-mongo_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。

4.1.2 RHEL8 で

MongoDBForeignDataWrapper 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%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. edb-as<xx>-mongo_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 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 MongoDB 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 MongoDBForeignDataWrapper をインストールします。

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

MongoDBForeignDataWrapper 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>-mongo_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 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 the MongoDB 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 MongoDBForeignDataWrapper をインストールします。

```
yum install edb-as<xx>-mongo_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。

4.1.4 CentOS8 で

MongoDBForeignDataWrapper 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:

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>-mongo_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 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 MongoDB 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 MongoDBForeignDataWrapper をインストールします。

```
dnf install edb-as<xx>-mongo_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 Debian または Ubuntu ホストへの MongoDB 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 MongoDBForeignDataWrapper 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 Debian パッケージをインストールします。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. EDB 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-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>-mongo-fdw
```

ここで、「xx」はサーババージョン番号です。

第 5 章

Features of the MongoDB Foreign Data Wrapper

MongoDB 外部データラッパーのキー機能は次のとおりです。

5.1 書き込み可能な FDW

MongoDB 外部データラッパーを使用すると、MongoDB サーバー上のデータを変更できます。ユーザーは、外部テーブルでローカルにデータを挿入、更新、削除する挿入により、リモート MongoDB コレクションのデータを INSERT、UPDATE および DELETE できます。こちらをご覧ください:

例:*MongoDB* 外部データラッパーの使用

データ型マッピング

5.2 接続プーリング

Mongo_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 自動化されたクリーンアップ

MongoDB 外部データラッパーを使用すると、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ている場合に特に役立ちます。DROP EXTENSION コマンドの構文は次の構文です。

```
DROP EXTENSION mongo_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 MongoDB Foreign Data Wrapper

MongoDB 外部データラッパーを使用する前に、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 MongoDBForeignDataWrapper 拡張機能を作成します。
2. *CREATESERVER* コマンドを使用して、MongoDB 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、MongoDB クラスターに存在するデータベースに対応する Postgres データベースのテーブルを定義します。

6.1 エクステンションを作成

CREATE EXTENSION コマンドを使用して、mongo_fdw 拡張機能を作成します。コマンドを呼び出すには、選択したクライアント（例、psql）を使用して、MongoDB サーバーのクエリを行う Postgres データベースに接続し、コマンドを呼び出します。

```
CREATE EXTENSION [IF NOT EXISTS] mongo_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、MongoDB 外部データラップをインストールします。

```
CREATE EXTENSION mongo_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 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 mongo_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プャーに対する 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 mongo_fdw 外部データラップャ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、外部サーバオブジェクトの接続情報を指定します。以下を含めることができます。

Option	Description
address	Mongo 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	MongoServer のポート番号。有効なレンジは 0~65535 です。デフォルト値は '27017' です。
authentication_database	ユーザが認証されるデータベース。このオプションは、パスワードベースの認証でのみ有効です。
replica_set	サーバーがメンバになっているレプリカセット。設定されている場合、ドライバは書き込み時にレプリカセットのプライマリを修正するために自動接続します。
read_preference	読み取り設定のオーダー。使用可能なオプションは、プライマリ [デフォルト]、二次的な、primaryPreferred、secondaryPreferred、および nearest です。
ssl	認証済みの暗号化された SSL 接続を要求します。デフォルトでは、値は false に設定されています。SSL を有効にするには、値を true に設定します。オプションについては、 http://mongoc.org/libmongoc/current/mongoc_ssl_opt_t.html を参照してください。
pem_file	SSL オプション
pem_pwd	SSL オプション
ca_file	SSL オプション
ca_dir	SSL オプション
crl_file	SSL オプション
weak_cert_validation	SSL オプション

Example

次のコマンドは、mongo_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 mongo_server 名前付けの外部サーバを作成します。

```
CREATE SERVER mongo_server FOREIGN DATA WRAPPER mongo_fdw OPTIONS (host '127.0.0.1',
↪port '27017');
```

外部サーバは、MongoDB 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（27017）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 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、MongoDB 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。

`username`: MongoDB 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。

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 mongo_server 名前付けのサーバに関連付けられます。

```
CREATE USER MAPPING FOR enterprisedb SERVER mongo_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 enterprisedb SERVER mongo_server OPTIONS (username 'mongo_user'
→, password 'mongo_pass');
```

このコマンドは、mongo_server という名前付けのサーバ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。MongoDB サーバーに接続すると、サーバーは mongo_user として認証し、パスワード mongo_pass 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 外部テーブルの作成

外部テーブルは、MongoDB 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、MongoDB サーバーに接続してコレクションを作成します。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、MongoDB ホストに存在するコレクション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、MongoDB 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 MongoDB 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

MongoDB 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 `BOOL` 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' [, ... ] ) ]`

MongoDB 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、MongoDB 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
database	クエリーするデータベースの名前。デフォルト値は test です。
collection	クエリーするコレクションの名前。デフォルト値は外部テーブル名前です。

Example

MongoDB サーバーに格納されているデータを使用するには、MongoDB コレクションの列を Postgres テーブルの列にマップするテーブルを 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 MongoDB のコレクションのための:

```
db.warehouse.find
(
  {
    "warehouse_id" : 1
  }
).pretty()
{
  "_id" : ObjectId("53720b1904864dc1f5a571a0"),
  "warehouse_id" : 1,
  "warehouse_name" : "UPS",
  "warehouse_created" : ISODate("2014-12-12T07:12:10Z")
}
```

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
(
  _id          NAME,
  warehouse_id INT,
  warehouse_name TEXT,
  warehouse_created TIMESTAMPTZ
)
SERVER mongo_server
OPTIONS (database 'db', collection 'warehouse');
```

テーブルの最初の列は、タイプ name の _id でなければなりません。

MongoDB の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: MongoDB 外部データラッパは、書き込み機能機能をサポートしています。

6.4.1 データ型マッピング

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

MongoDB (BSON Type)	Postgres
ARRAY	JSON
BOOL	BOOL
BINARY	BYTEA
DATE_TIME	DATE/TIMESTAMP/TIMESTAMPTZ
DOCUMENT	JSON
DOUBLE	FLOAT/FLOAT4/FLOAT8/DOUBLEPRECISION/NUMERIC
INT32	SMALLINT/INT2/INT/INTEGER/INT4
INT64	BIGINT/INT8
OID	NAME
UTF8	BPCHAR/VARCHAR/CHARCTERVARYING/TEXT

第 7 章

Example: Using the MongoDB Foreign Data Wrapper

MongoDB 外部データラップを使用する前に、クライアントアプリケーションを使用してデータベースに接続する必要があります。次の例は、psql クライアントでラッパーを使用する方法を示しています。psql に接続したら、次の例の手順を実行できます。

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

-- create server object
CREATE SERVER mongo_server
    FOREIGN DATA WRAPPER mongo_fdw
    OPTIONS (address '127.0.0.1', port '27017');

-- create user mapping
CREATE USER MAPPING FOR postgres
    SERVER mongo_server
    OPTIONS (username 'mongo_user', password 'mongo_pass');

-- create foreign table
CREATE FOREIGN TABLE warehouse
    (
        _id name,
        warehouse_id int,
        warehouse_name text,
        warehouse_created timestampz
    )
    SERVER mongo_server
    OPTIONS (database 'db', collection 'warehouse');

-- Note: first column of the table must be "_id" of type "name".

-- select from table
```

(次のページに続く)

(前のページからの続き)

```

SELECT * FROM warehouse WHERE warehouse_id = 1;
-----+-----+-----+-----+
      _id      | warehouse_id | warehouse_name | warehouse_created
-----+-----+-----+-----+
 53720b1904864dc1f5a571a0 |          1 | UPS           | 2014-12-12 12:42:10+05:30
(1 row)

db.warehouse.find
(
  {
    "warehouse_id" : 1
  }
).pretty()
{
  "_id" : ObjectId("53720b1904864dc1f5a571a0"),
  "warehouse_id" : 1,
  "warehouse_name" : "UPS",
  "warehouse_created" : ISODate("2014-12-12T07:12:10Z")
}

-- insert row in table
INSERT INTO warehouse VALUES (0, 2, 'Laptop', '2015-11-11T08:13:10Z');

db.warehouse.insert
(
  {
    "warehouse_id" : NumberInt(2),
    "warehouse_name" : "Laptop",
    "warehouse_created" : ISODate("2015-11-11T08:13:10Z")
  }
)

-- delete row from table
DELETE FROM warehouse WHERE warehouse_id = 2;

db.warehouse.remove
(
  {
    "warehouse_id" : 2
  }
)

-- update a row of table
UPDATE warehouse SET warehouse_name = 'UPS_NEW' WHERE warehouse_id = 1;

db.warehouse.update
(
  {
    "warehouse_id" : 1
  }

```

(次のページに続く)

(前のページからの続き)

```
    },
    {
        "warehouse_id" : 1,
        "warehouse_name" : "UPS_NEW",
        "warehouse_created" : ISODate("2014-12-12T07:12:10Z")
    }
)

-- explain a table
EXPLAIN SELECT * FROM warehouse WHERE warehouse_id = 1;
          QUERY PLAN
-----
Foreign Scan on warehouse  (cost=0.00..0.00 rows=1000 width=84)
  Filter: (warehouse_id = 1)
  Foreign Namespace: db.warehouse
(3 rows)

-- collect data distribution statistics
ANALYZE warehouse;
```


第 8 章

Identifying the MongoDB Foreign Data Wrapper Version

MongoDB 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 mongo_fdw_version();
```

ファンクションはバージョン番号を返します。

```
mongo_fdw_version
-----
<xxxxxx>
```


第 9 章

Conclusion

MongoDB 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, 39

Configuring the MongoDB Foreign Data Wrapper, 23

Example: Using the MongoDB Foreign Data Wrapper, 33

Features of the MongoDB Foreign Data Wrapper, 21

Identifying the MongoDB Foreign Data Wrapper Version, 37

Installing MongoDB Data Adapter on Debian, 18

Installing MongoDB Data Adapter on Ubuntu, 18

Installing the MongoDB Foreign Data Wrapper, 9

Requirements Overview, 5

rhel8, 12

supported platforms, 5

supported server versions, 5

What's New, 3