



EDB

EDB ODBC Connector

リリース 12.2.0.2

EDB

2022 年 05 月 27 日

目次

第 1 章	Release notes	3
1.1	Version 13.02.0000.01	3
1.2	Version 13.01.0000.02	3
1.3	Version 13.01.0000.01	4
1.4	Version 13.00.0000.01	4
第 2 章	Product compatibility	5
第 3 章	Installing the EDB ODBC Connector	7
3.1	Installing on Linux	7
3.1.1	Installing EDB ODBC Connector on IBM Power (ppc64le)	8
3.1.2	Installing EDB ODBC Connector on Linux x86 (amd64)	11
3.2	Installing the ODBC Connector on Windows	20
第 4 章	Creating a data source	27
第 5 章	Connection properties	29
5.1	Windows でのデータソース定義の追加	36
5.2	Linux でのデータソース定義の追加	41
第 6 章	Driver functionality	45
6.1	SQLGetInfo ()	45
6.2	接続属性	61
6.2.1	SQLGetConnectAttr ()	61
6.2.2	SQLSetConnectAttr ()	62
6.3	環境属性	64
6.3.1	SQLGetEnvAttr ()	64
6.4	SQLSetEnvAttr ()	65
6.5	ステートメント属性	66
6.5.1	SQLGetStmtAttr ()	66
6.5.2	SQLSetStmtAttr ()	68
6.6	エラーハンドリング	68
6.6.1	SQLGetDiagRec ()	69
6.7	サポートされている ODBC API 関数	70
6.8	サポートされているデータ型	72
6.9	ADO ユーザーの前提条件	72
6.10	スレッドセーフ	72
第 7 章	Scram compatibility	73
第 8 章	SLES 15 on IBM Power (ppc64le)	75
8.1	リポジトリのセットアップ	75

8.2 パッケージのインストール	76
第 9 章 Upgrading a Linux installation	77

ODBC (Open Database Connectivity) は、クライアントアプリケーションが ODBC ドライバを提供する任意のデータベースに接続できるようにするプログラミングインタフェースです。EDB ODBC コネクタは、ODBC 準拠のクライアントアプリケーションが EDB Postgres Advanced Server データベースに接続できるようにするインタフェースです。EDB ODBC コネクタを使用すると、他のデータベースと連携するように設計されたアプリケーションを EDB Postgres Advanced Server で実行できます。ODBC コネクタは、クライアントアプリケーションが接続を確立し、クエリを送信し、EDB Postgres Advanced Server から結果を取得する方法を提供します。

ODBC コネクタはアプリケーションの移植性を提供しレベルが、移植性は制限されています。接続を提供しますが、コマンドの互換性を保証するものではありません。別のデータベースで受け入れ可能なコマンドは、EDB Postgres Advanced Server では機能しない場合があります。

一般的な ODBC アプリケーションのメジャーコンポーネントは次のとおりです。

ODBC のバインディングを持つ言語で記述されたクライアントアプリケーション Windows または Linux の名前付け接続を処理する ODBC アドミニストレーター - データベース固有の ODBC ドライバ (ODBC コネクタ) ODBC 準拠サーバー (EDB Postgres Advanced Server)

クライアントアプリケーションは、ODBC にバインドされている任意の言語で作成できます。C、MS-Access、および C++ はほんの一部です。

第1章 Release notes

ODBC JDBC コネクタの文書には、ODBC JDBC コネクタのバージョン 13 が記載されています。

リリースノートでは、リリースの新機能について説明します。マイナーリリースまたはパッチリリースで新しい機能が導入されると、コンテンツ内のインジケータは、新しい機能を導入したバージョンを識別します。

Version	Release date
<i>Version 13.02.0000.01</i>	2022 May 17
<i>Version 13.01.0000.02</i>	2021 Dec 12
<i>Version 13.01.0000.01</i>	2021 Sep 14
<i>Version 13.00.0000.01</i>	2020 Nov 19

1.1 Version 13.02.0000.01

EDB ODBC Connector 13.02.0000.01 には、次の機能強化が含まれています。

Type	Description
Upstream merge	アップストリームコミュニティドライババージョン 13.02.0000 と統合されました。詳細については、コミュニティ :ref:`Release notes<Release notes>` を参照してください。
Bug fix	データベースの MS SQL リンクサーバーの繰り返し [サポートチケット # 1044249]。

1.2 Version 13.01.0000.02

EDB ODBC コネクタ 13.01.0000.02 には、次の機能強化が含まれています。

Type	Description
Enhancement	EDB Postgres Advanced Server バージョン 14.1.0 のサポート。

1.3 Version 13.01.0000.01

EDB ODBC Connector 13.01.0000.01 には、次のアップストリームマージが含まれています。

Type	Description
Upstream merge	アップストリームコミュニティドライババージョン 13.01.0000 と統合されました。詳細については、コミュニティ :ref:`Release notes<Release notes>` を参照してください。

1.4 Version 13.00.0000.01

EDB ODBC Connector 13.00.0000.01 には、次のアップストリームマージが含まれています。

Type	Description
Upstream merge	アップストリームコミュニティドライババージョン 13.00.0000 と統合されました。詳細については、コミュニティ :ref:`Release notes<Release notes>` を参照してください。

第2章 Product compatibility

この表には、最新の ODBC コネクタバージョンと、対応する EDB Postgres Advanced Server (EPAS) バージョンがリストされています。詳細については、[Product compatibility](#) を参照してください。

ODBC Connector	EPAS 14	EPAS 13	EPAS 12	EPAS 11	EPAS 10
13.01.0.02	Y	Y	Y	Y	Y
13.01.0.01	N	Y	Y	Y	Y
13.00.0.01	N	Y	Y	Y	Y

第3章 Installing the EDB ODBC Connector

EDB ODBC コネクタは次の場所にインストールできます。

- *Installing on Linux*
- *Installing the ODBC Connector on Windows*

3.1 Installing on Linux

EDB ODBC コネクタをインストールするには、EnterpriseDB 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。EnterpriseDB リポジトリへのアクセスを許可する資格情報を要求するには、[EDB Repository Access instructions](#) を参照してください。

プラットフォーム固有のインストール手順については、以下を参照してください。

- Linux x86-64 (amd64) :
 - *Installing EDB ODBC Connector on RHEL 8/OL 8 x86*
 - *Installing EDB ODBC Connector on Rocky Linux 8/AlmaLinux 8 x86*
 - *Installing EDB ODBC Connector on RHEL 7/OL 7 x86*
 - *Installing EDB ODBC Connector on CentOS 7 x86*
 - *Installing EDB ODBC Connector on SLES 15 x86*
 - *Installing EDB ODBC Connector on SLES 12 IBM Power (ppc64le)*
 - *Installing EDB ODBC Connector on Ubuntu 20.04/Debian 10 x86*
 - *Installing EDB ODBC Connector on Ubuntu 18.04/Debian 9 x86*
- Linux on IBM Power (ppc64le) :
 - *Installing EDB ODBC Connector on RHEL 8 IBM Power (ppc64le)*
 - *Installing EDB ODBC Connector on RHEL 7 IBM Power (ppc64le)*
 - *Installing EDB ODBC Connector on SLES 12 IBM Power (ppc64le)*

3.1.1 Installing EDB ODBC Connector on IBM Power (ppc64le)

オペレーティングシステム固有のインストール手順については、以下を参照してください。

- [Installing EDB ODBC Connector on RHEL 8 IBM Power \(ppc64le\)](#)
- [Installing EDB ODBC Connector on RHEL 7 IBM Power \(ppc64le\)](#)
- [Installing EDB ODBC Connector on SLES 12 IBM Power \(ppc64le\)](#)

Installing EDB ODBC Connector on RHEL 8 IBM Power (ppc64le)

インストールを完了するには2つの手順があります。

- リポジトリのセットアップ
- パッケージのインストール

各ステップで、スーパーユーザとしてログインする必要があります。

スーパーユーザとしてログするには：

```
sudo su -
```

リポジトリのセットアップ

1. EDB に登録して EDB リポジトリの資格情報を受け取るには、[Repository Access Request](#) にアクセスしてください。
2. EDB リポジトリをセットアップします。

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

これにより、`/etc/yum.repos.d/edb.repo` 構成ファイルが作成されます。

1. EDB 認証情報を `edb.repo` ファイルに追加します。

```
sed -i "s<username>:<password>@USERNAME:PASSWORD@" /etc/yum.repos.d/edb.repo
```

`USERNAME:PASSWORD` は、[EDB account](#) から入手できるユーザー名とパスワードです。

1. EPEL リポジトリをインストールし、キャッシュをリフレッシュします。

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

1. EPEL パッケージソフトはそこからのパッケージソフトに依存する可能性があるため、`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"
```

1. ビルトインの PostgreSQL モジュールを無効にします：edb_notranlate_5

パッケージのインストール

```
dnf -y install edb-odbc
```

Installing EDB ODBC Connector on RHEL 7 IBM Power (ppc64le)

ODBC コネクタをインストールする前に、以下の前提条件パッケージソフトをインストールし、EDB : 1 から資格情報を要求する必要があります。Advance Toolchain をインストールするには、次のコマンドを使用します：edb_notranlate_0

2. 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. リポジトリ設定ファイルを作成します。ファイルを変更し、ユーザ名とパスワードを入力します。ODBC コネクタをインストールします。

リポジトリ設定ファイルの作成

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。

ファイルの変更、ユーザ名前とパスワードの提供

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

ODBC コネクタのインストール

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 ODBC Connector をインストールします。

```
yum install edb-odbc

yum install edb-odbc-devel
```

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、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。

Installing EDB ODBC Connector on SLES 12 IBM Power (ppc64le)

インストールを完了するには 2 つの手順があります。

- リポジトリのセットアップ
- パッケージのインストール

各ステップで、スーパーユーザとしてログインする必要があります。

```
# To log in as a superuser:
sudo su -
```

リポジトリを設定する前に、EDB に登録する必要があります。EDB リポジトリの資格情報を受信するには、[Repository Access Request](#) にアクセスしてください。

リポジトリのセットアップ

リポジトリの設定は1回限りのタスクです。リポジトリをすでに設定している場合、これらの手順を実行する必要はありません。

```
# Install the repository configuration and enter your EDB repository
# credentials when prompted
zypper addrepo https://zypp.enterprisedb.com/suse/edb-sles.repo

# Install SUSEConnect to register the host with SUSE, allowing access to
# SUSE repositories
zypper install SUSEConnect

# Register the host with SUSE, allowing access to SUSE repositories
# Replace REGISTRATION_CODE and EMAIL with your SUSE registration
# information
SUSEConnect -r REGISTRATION_CODE -e EMAIL

# Activate the required SUSE modules
SUSEConnect -p PackageHub/12.5/ppc64le
SUSEConnect -p sle-sdk/12.5/ppc64le

# Refresh the metadata
zypper refresh
```

パッケージのインストール

```
zypper -n install edb-odbc
```

3.1.2 Installing EDB ODBC Connector on Linux x86 (amd64)

オペレーティングシステム固有のインストール手順については、以下を参照してください。

- *Installing EDB ODBC Connector on RHEL 8/OL 8 x86*
- *Installing EDB ODBC Connector on Rocky Linux 8/AlmaLinux 8 x86*
- *Installing EDB ODBC Connector on RHEL 7/OL 7 x86*
- *Installing EDB ODBC Connector on CentOS 7 x86*
- *Installing EDB ODBC Connector on SLES 15 x86*
- *Installing EDB ODBC Connector on SLES 12 IBM Power (ppc64le)*
- *Installing EDB ODBC Connector on Ubuntu 20.04/Debian 10 x86*
- *Installing EDB ODBC Connector on Ubuntu 18.04/Debian 9 x86*

Installing EDB ODBC Connector on RHEL 8/OL 8 x86

ODBC コネクタ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 から資格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、次のことができます。

1. リポジトリ設定ファイルを作成します。ファイルを変更し、ユーザ名とパスワードを入力します。edb-odbc をインストールします。

リポジトリ設定ファイルの作成

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。

ファイルの変更、ユーザ名とパスワードの提供

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

ODBC コネクタのインストール

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 ODBC コネクタをインストールします。

```
dnf install edb-odbc

dnf install edb-odbc-devel
```

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、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。

Installing EDB ODBC Connector on Rocky Linux 8/AlmaLinux 8 x86

ODBC コネクタ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 から資格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%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. リポジトリ設定ファイルを作成します。ファイルを変更し、ユーザ名前とパスワードを入力します。edb-odbc をインストールします。

リポジトリ設定ファイルの作成

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。

ファイルの変更、ユーザ名前とパスワードの提供

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

ODBC コネクタのインストール

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 ODBC Connector をインストールします。

```
dnf install edb-odbc

dnf install edb-odbc-devel
```

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、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。

Installing EDB ODBC Connector on RHEL 7/OL 7 x86

ODBC コネクタ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 から資格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. リポジトリ設定ファイルを作成します。ファイルを変更し、ユーザ名前とパスワードを入力します。
`edb-odbc` をインストールします。

リポジトリ設定ファイルの作成

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。

ファイルの変更、ユーザ名前とパスワードの提供

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

ODBC コネクタのインストール

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 ODBC Connector をインストールします。

```
yum install edb-odbc

yum install edb-odbc-devel
```

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、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。

</div>

Installing EDB ODBC Connector on CentOS 7 x86

ODBC コネクタ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 から資格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. リポジトリ設定ファイルを作成します。ファイルを変更し、ユーザ名とパスワードを入力します。**edb-odbc** をインストールします。

リポジトリ設定ファイルの作成

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。

ファイルの変更、ユーザ名とパスワードの提供

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
```

(次のページに続く)

(前のページからの続き)

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

ODBC コネクタのインストール

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 ODBC Connector をインストールします。

```
yum install edb-odbc

yum install edb-odbc-devel
```

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、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。

Installing EDB ODBC Connector on SLES 15 x86

インストールを完了するには2つの手順があります。

- リポジトリのセットアップ
- パッケージのインストール

各ステップで、スーパーユーザとしてログインする必要があります。

```
# To log in as a superuser:
sudo su -
```

リポジトリを設定する前に、EDB に登録する必要があります。EDB リポジトリの資格情報を受信するには、[Repository Access Request](#) にアクセスしてください。

リポジトリのセットアップ

リポジトリの設定は1回限りのタスクです。リポジトリをすでに設定している場合、これらの手順を実行する必要はありません。

```
# Install the repository configuration and enter your EDB repository
# credentials when prompted
zypper addrepo https://zypp.enterprisedb.com/suse/edb-sles.repo
```

(次のページに続く)

```
# Install SUSEConnect to register the host with SUSE, allowing access to
# SUSE repositories
zypper install SUSEConnect

# Register the host with SUSE, allowing access to SUSE repositories
# Replace REGISTRATION_CODE and EMAIL with your SUSE registration
# information
SUSEConnect -r REGISTRATION_CODE -e EMAIL

# Activate the required SUSE module
SUSEConnect -p PackageHub/15.3/x86_64

# Refresh the metadata
zypper refresh
```

パッケージのインストール

```
zypper -n install edb-odbc
```

Installing EDB ODBC Connector on SLES 12 x86

zypper パッケージマネージャを使用して、SLES 12 ホストにコネクタをインストールできます。zypper は、パッケージをインストールするときにパッケージの依存関係を満たそうとしますが、EDB でホストされていない特定のリポジトリへのアクセスが必要です。コネクタをインストールする前に、次のコマンドを使用して EDB リポジトリ構成ファイルを SLES ホストに追加します。

```
zypper addrepo https://zypp.enterprisedb.com/suse/edb-sles.repo
```

リポジトリ構成ファイルを作成した後、`zypper refresh` コマンドを使用して SLES ホストのメタデータをリフレッシュし、EDB リポジトリを組み込みます。

User Name および Password の入力を求められたら、EDB リポジトリの接続資格情報を入力します。リポジトリの資格情報を要求するには、[the EDB website](#) にアクセスしてください。

EDB Postgres Advanced Server またはサポートコンポーネントをインストールする前に、SUSEConnect および SUSE Package Hub 拡張を SLES ホストに追加し、ホストを SUSE に登録して、SUSE リポジトリへのアクセスを許可する必要があります。次のコマンドを使用します。

```
zypper install SUSEConnect SUSEConnect -r 'REGISTRATION_CODE' -e 'EMAIL' SUSEConnect -p
PackageHub/12.4/x86_64 SUSEConnect -p sle-sdk/12.4/x86_64
```

SUSE ホストの登録の詳細については、[SUSE website](#) にアクセスしてください。

次に、zypper ユーティリティを使用してコネクタをインストールできます。

```
zypper install edb-odbc
```

```
zypper install edb-odbc-devel
```

Installing EDB ODBC Connector on Ubuntu 20.04/Debian 10 x86

Debian または Ubuntu ホストに DEB パッケージ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要求するには、[EDB website](#) にアクセスしてください。

次の手順では、EDB apt 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. EDB リポジトリを構成します。

3. EDB リポジトリをセットアップします。

```
-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
```

2. 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。

```
-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 署名キーを追加します。

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

5. リポジトリメタデータを更新します。

```
apt-get update
```

6. DEB パッケージをインストールします。

```
apt-get install edb-odbc
apt-get install edb-odbc-dev
```

Installing EDB ODBC Connector on Ubuntu 18.04/Debian 9 x86

Debian または Ubuntu ホストに DEB パッケージ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要求するには、[EDB website](#) にアクセスしてください。

次の手順では、EDB apt 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. EDB リポジトリを構成します。

edb_notranlate_1 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 署名キーを追加します。

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

5. リポジトリメタデータを更新します。

```
apt-get update
```

6. DEB パッケージをインストールします。

```
apt-get install edb-odbc  
apt-get install edb-odbc-dev
```

3.2 Installing the ODBC Connector on Windows

EDB コネクタインストールウィザードを使用して、システムに ODBC コネクタを追加できます。ウィザードは [EDB website](#) で利用可能です。

インストーラをダウンロードし、インストーラアイコンを右クリックして、コンテキストメニューから **Run As Administrator** を選択します。

Language Selection ポップアップが開いたら、インストール言語を選択し、OK をクリックして Setup ウィンドウに進みます（下図を参照）。

ODBC コネクタインストールウィザード。

Next をクリックして続行します。

インストールダイアログ

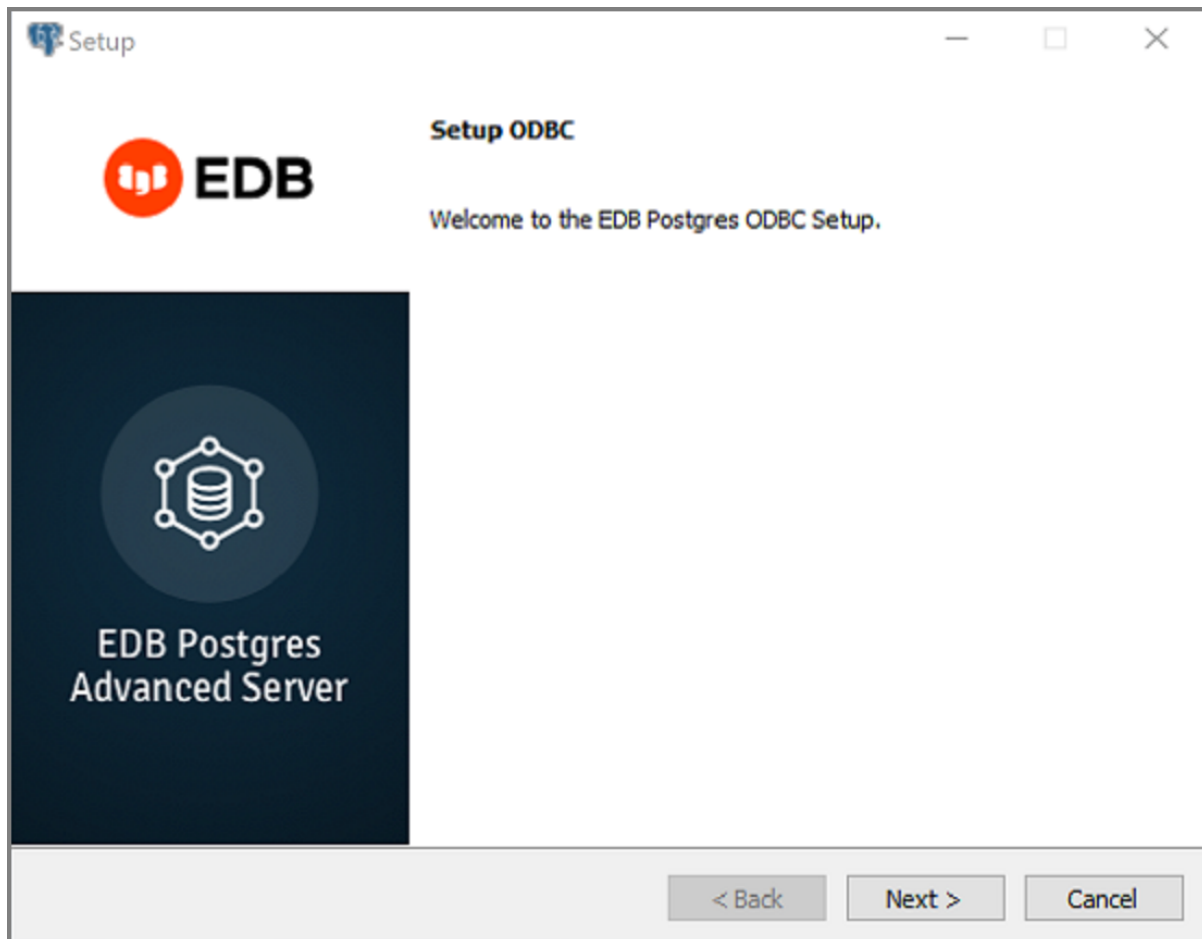


図 1: The ODBC Connectors Installation wizard.

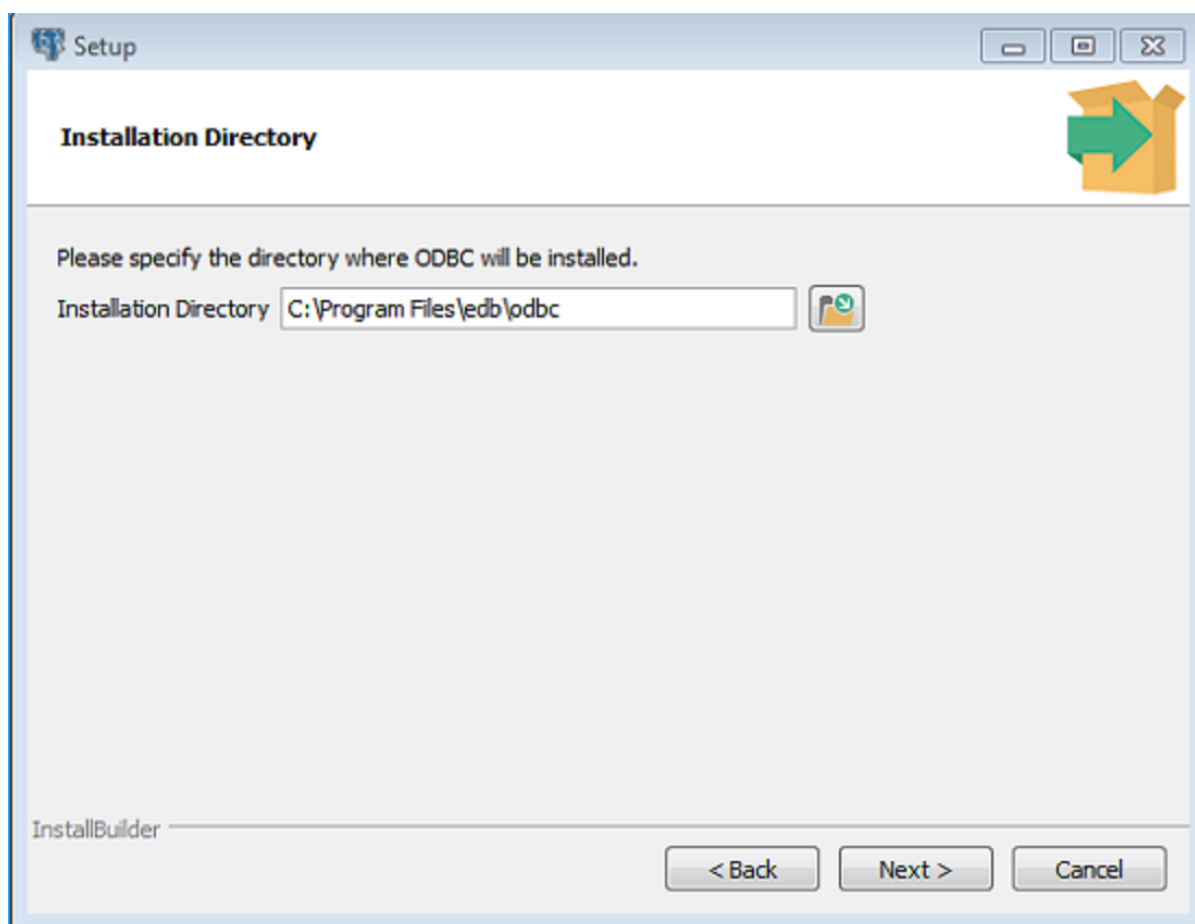


図 2: The Installation dialog.

Installation Directory ダイアログを使用して、コネクタをインストールするディレクトリを指定し、Next をクリックして続行します。

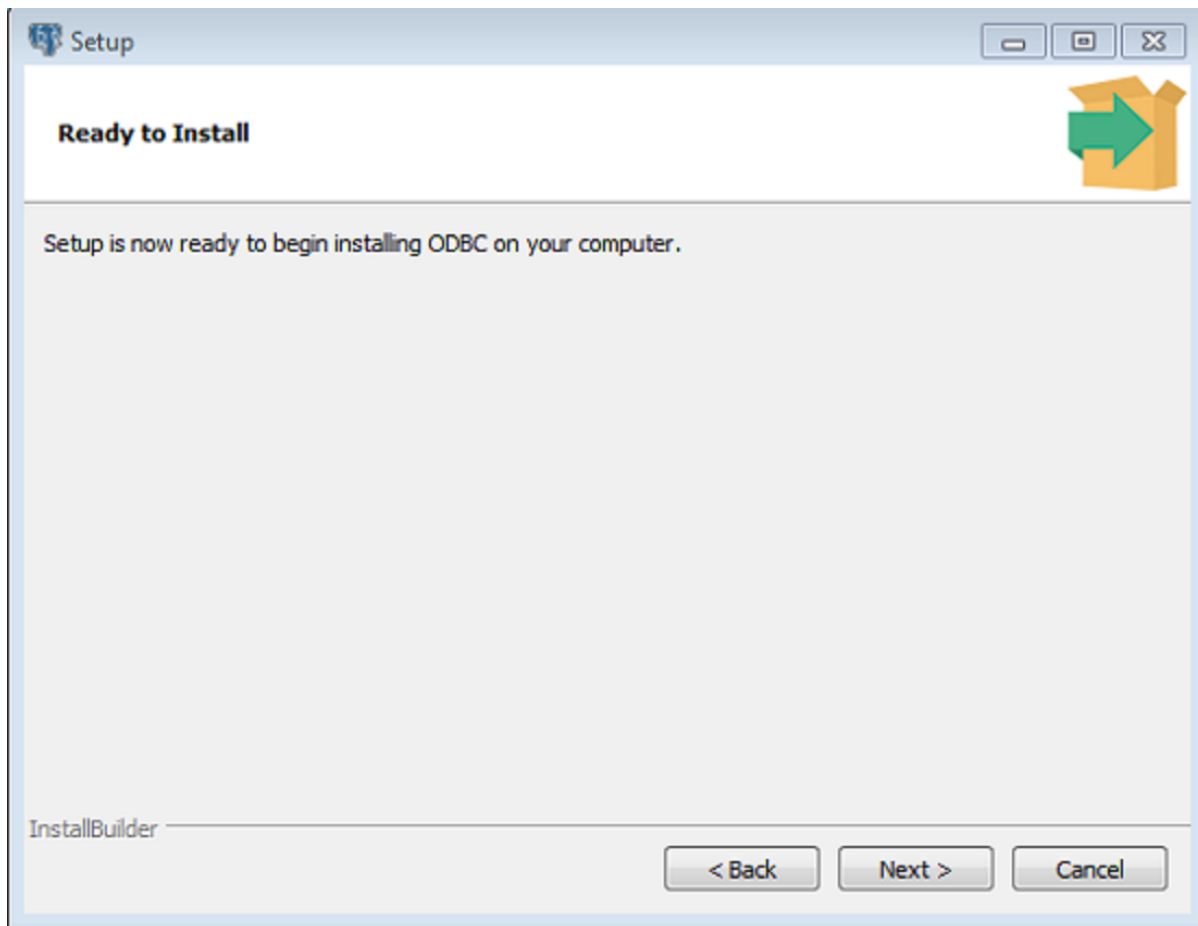


図 3: The Ready to Install dialog.

[インストールの準備完了] ダイアログ。

Ready to Install ダイアログで Next をクリックして、インストールをスタートします。ポップアップダイアログがインストールウィザードの進行状況を確認します。

インストールが完了しました。

ウィザードがセットアップの完了を通知したら、Finish ボタンをクリックしてダイアログを終了します。

StackBuilder Plus を使用して、既存の Advanced Server インストールでコネクタを追加または更新することもできます。StackBuilder Plus を開くには、Windows Apps メニューから StackBuilder Plus を選択します。

StackBuilder Plus の起動

StackBuilder Plus が開いたら、画面の指示に従います。ツリーコントロールの Database Drivers ノードから EnterpriseDB ODBC Connector オプションを選択します。

コネクタインストーラの選択。

画面上のウィザードの指示に従って、EDB コネクタのインストールを追加または更新します。

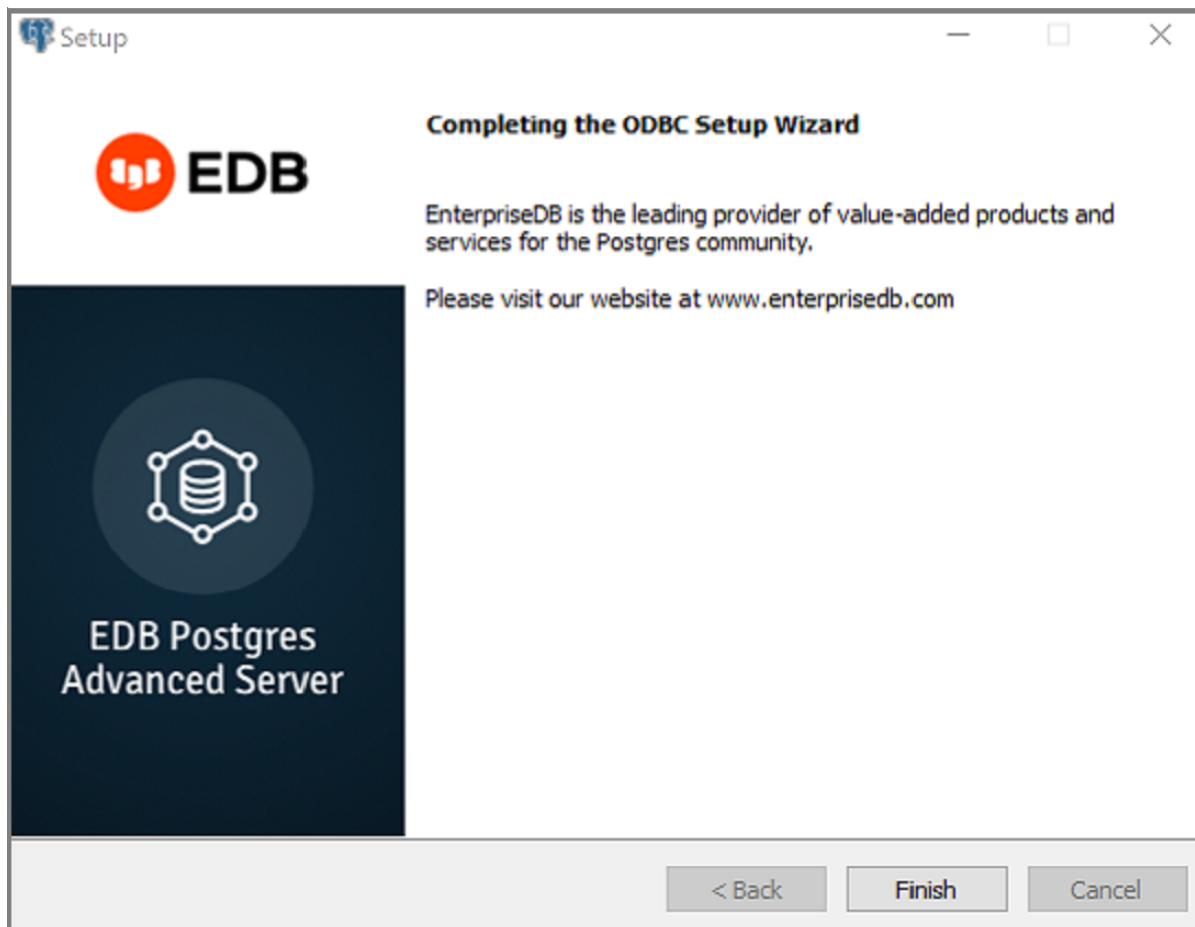


図 4: The installation is complete.

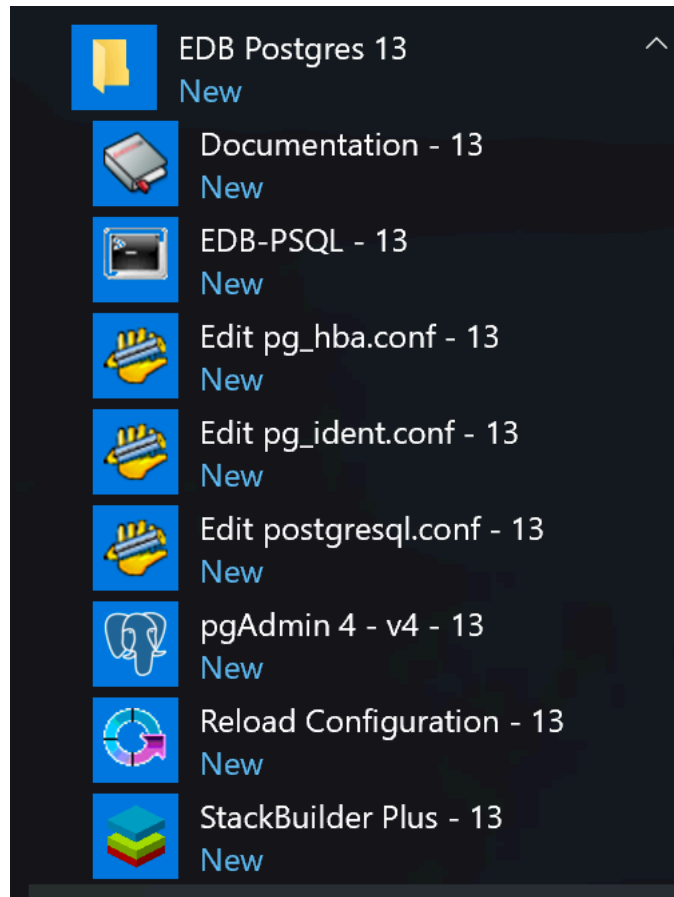


図 5: Starting StackBuilder Plus

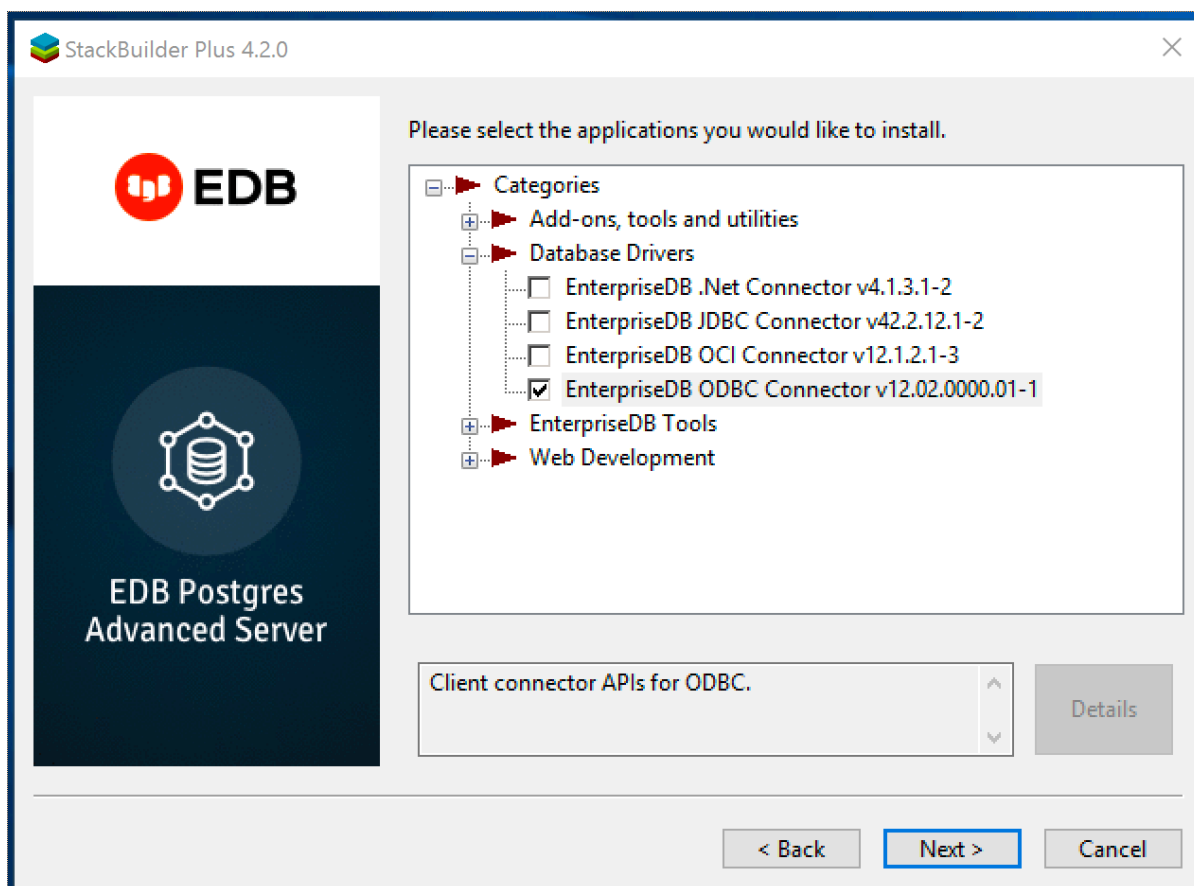


図 6: Selecting the Connectors installer.

第4章 Creating a data source

クライアントアプリケーションがサーバーとの接続を確立しようとする、通常、データソース名前 (DSN) が提供されます。ドライバマネージャは、アプリケーションによって提供される DSN と一致する名前を持つデータソースの ODBC 構成データベースを検索します。

Linux または Unix ホストでは、データソースは通常 `/etc/odbc.ini` 名前付けのファイルで定義されます。名前と場所は異なる場合があります。次のコマンドを使用して、`unixODBC` がデータソース定義を検索している場所を見つけます。

```
$ odbc_config --odbcini --odbcinstini
```

Windows ホストでは、通常、データソースは Windows レジストリで定義されます。

ファイル DSN と呼ばれるデータソース定義をプレーンテキストファイルに保存することもできます。EDB-ODBC ドライバの一般的なデータソース定義は次のようになります。

```
$ cat /etc/odbc.ini
[EnterpriseDB]
Description = EnterpriseDB DSN
Driver = EnterpriseDB
Trace = yes
TraceFile = /tmp/odbc.log
Database = edb
Servername = localhost
Username = enterprisedb
Password = manager
Port = 5444
```

データソースの最初の行はデータソース名前です。名前は、大括弧で囲まれた一意の識別子です。データソース名前の後には、データソースを `make` する個々の接続プロパティを識別するシリーズの `keyword=value` ペアが続きます。

ODBC アドミニストレーターユーティリティは、ODBC 接続用の名前付けデータソースを作成します。ほとんどの場合、ODBC Administrator ユーティリティはオペレーティングシステムに付属しています。Windows または `unixODBC` を使用している場合、このツールは ODBC データソースアドミニストレーターと呼ばれます。オペレーティングシステムに ODBC アドミニストレーターが含まれていない場合は、サードパーティのオプションをオンラインで利用できます。

[Adding a data source definition in Windows](#) および [Linux でのデータソース定義の追加](#) では、各オペレーティングシステムで使用可能グラフィカルツールを使用して、Windows および Linux でデータソースを追加する方法を説明します。データソースを定義するプロセス中に、接続プロパティのセットを指定するよう求められます。

EDB-ODBC connection properties には、オプションのデータソース接続プロパティに関する情報が含まれています。グラフィカルツールで接続プロパティを指定するか、テキストエディタで

odbc.ini ファイルを編集できます。

第5章 Connection properties

次の表に、グラフィカル接続マネージャツールのダイアログボックスまたは名前付けデータソースを定義する `odbc.ini` ファイルで指定できる接続プロパティを示します。列は、接続プロパティ (ODBC Administrator ダイアログに表示される)、対応するキーワード (`odbc.ini` ファイルに表示される)、プロパティのデフォルト値、および接続プロパティの説明を識別します。

Property	Keyword name	Default value	Description
Database	Database	None	接続するデータベースの名前。
Driver	Driver	EDB-ODBC	ODBC ドライバの名前。
Server	Servename	Localhost	接続しているサーバーの名前または IP アドレス。
dbms_name	dbms_name	EnterpriseDB	データベースシステム。 EnterpriseDB または PostgreSQL のいずれか。
Description	Description		データソースの説明的な名前。
User Name	Username		このデータソースがサーバーへの接続に使用するユーザの名前。
Password	Password		この名前付けデータソースに関連付けられているユーザのパスワード。
CPOutetimeout	CPOutetimeout	0	接続がタイムアウトするまでの秒数 (コネクションプーリング環境)。
Port	Port	5444	ポストマスターがリスンしている TCP ポート。
Protocol	Protocol	7.4	指定されている場合、ドライバは指定されたプロトコルバージョンを強制的に使用します。
Level of Rollback on Errors	Use the <i>Protocol</i> option to specify rollback behavior.	Transaction Level	ドライバがエラーを処理する方法を指定します。 0-ロールバックしない 1-トランザクションをロールバックする 2-ステートメントをロールバックする
Usage Count	UsageCount	1	このドライバを使用したインストールの数。
Read Only	ReadOnly	No	接続が読み取り専用であることを指定します。

次のページに続く

表 1 – 前のページからの続き

Property	Keyword name	Default value	Description
Show System Tables	ShowSystemTables	No	有効にすると、ドライバは `SQLTables()` フังก์ションの結果セットでシステムテーブルを報告します。
OID Options: Show Column	ShowOidColumn	No	有効にすると、`SQLColumns()` フังก์ションは OID 列を報告します。
OID Options: Fake Index	FakeOidIndex	No	有効にすると、`SQLStatistics()` フังก์ションは、各 OID 列に一意インデックスが存在することを報告します。
Keyset Query Optimization	Ksqr	On	有効にすると、キーセットクエリ (MS Jet データベースエンジンによって生成される) のサーバサイドのサポートが強制されます。
Recognize Unique Indexes	UniqueIndex	On	有効にすると、`SQLStatistics()` フังก์ションは一意性インデックスを報告します。有効になっていない場合、`SQLStatistics()` フังก์ションは、インデックスが重複値を許可することを報告します。
Use Declare/Fetch	UseDeclareFetch	Off	有効にすると、ドライバはサーバサイドカーソルを使用します。`UseDeclareFetch` を有効にするには、`1` の値を指定します。`UseDeclareFetch` を無効にするには、`0` の値を指定します。
CommLog	CommLog	Off	有効にすると、すべてのクライアント/サーバートラフィックをログファイルに記録します。
Parse Statements	Parse	Off	有効にすると、ドライバは、`SQLNumResultCols()`、`SQLDescribeCol()`、または `SQLColAttributes()` 関数を呼び出すときに、シンプル `SELECT` ステートメントを解析します。
Cancel as FreeStmt	CancelAsFreeStmt	Off	有効にすると、`SQLCancel()` フังก์ションがユーザーに代わって `SQLFreeStmt(SQL_Close)` を呼び出します。

次のページに続く

表 1 – 前のページからの続き

Property	Keyword name	Default value	Description
MyLog	Debug	Off	有効にすると、ドライバはその作業をログファイルに記録します。Windows では、ファイル名前は `C:\mylog <>>` <process-id>` です。Linux では、ファイル名前は /tmp/ <ref>:mylog <>>` <username><process-id>.log` です。
Unknown Sizes	UnknownSizes	Maximum	`SQLDescribeCol()` および `SQLColAttributes()` 関数が列のサイズを計算する方法を決定します。 `0` を指定して、そのタイプに許可されている最大サイズをドライバにレポートさせます。 `1` を指定すると、ドライバに不明な長さをレポートさせ、`2` を指定すると、ドライバに結果セットを検索させて最長値を検出させます。 `UseDeclareFetch` を有効にした場合は、`2` を指定しないでください。
Text as LongVarchar	TextAsLongVarChar	8190	有効にすると、ドライバは TEXT 列を `SQL_LONGVARCHAR` 型であるかのように扱います。無効にすると、ドライバは TEXT 列を `SQL_VARCHAR` 値として扱います。
Unknown as Long Varchar	LongVarChar	False	有効にすると、ドライバは未知のタイプの値を `SQL_LONGVARCHAR` 値として扱います。無効にすると、ドライバは未知のタイプの値を `SQL_VARCHAR` 値として扱います。デフォルトでは、不明なタイプの値は `Y` 値として扱われます。
Bools as Char	BoolsAsChar	On	有効にすると、ドライバは BOOL 列を `SQL_CHAR` 値として扱います。無効にすると、BOOL 列は `SQL_BIT` 値として扱われます。

次のページに続く

表 1 – 前のページからの続き

Property	Keyword name	Default value	Description
Max Varchar	MaxVarcharSize	255	有効にすると、ドライバは`MaxVarCharSize`よりも長い`VARCHAR`および`BPCHAR`の値を`SQL_LONGVARCHAR`の値として扱います
Max Long Varchar Size	MaxLongVarchar-Size	8190	`TextAsLongVarChar`がオンの場合、ドライバは TEXT 値が`MaxLongVarcharSize`バイト長であることを報告します。 `UnknownAsLongVarChar`がオンの場合、不明なタイプの列は`MaxLongVarcharSize`バイト長です。それ以外の場合、それらは`MaxVarcharSize`バイトの長さであると報告されます。
Cache Size	Fetch	100	`UseDeclareFetch`が有効な場合にドライバがフェッチする行数を決定します。
SysTable Prefixes	ExtraSysTablePre-fixes	<i>Windows</i> でのデータソース定義の追加;	<i>SysTablePrefixes</i> フィールドを使用して、テーブルがシステムテーブルであることを示すプレフィックスのセミコロン区切りリストを指定します。デフォルトでは、リストには <i>Windows</i> でのデータソース定義の追加; が含まれています。
Cumulative Row Count for Insert	MapSqlParcNo-Batch	Off/0	有効にすると、`SQLRowCount()`関数は、`INSERT`ステートメントのパラメータ設定の配列全体について、単一の累積行カウントを返します。無効にすると、パラメータ設定ごとに個別の行カウントが返されます。デフォルトでは、このオプションは無効になっています。
LF$\rightarrow$ CR/LF conversion	LFConversion	System Dependent	LF <math>\rightarrow</math> CR/LF 変換オプションは、サーバーから文字値を取得するときに改行文字を復帰/改行ペアに変換し、文字値を送信するときに改行文字に戻すようにドライバに指示しますサーバー。デフォルトでは、このオプションは有効になっています。

次のページに続く

表 1 – 前のページからの続き

Property	Keyword name	Default value	Description
Updatable Cursors	UpdatableCursors	Off	`SQLSetPos()`または`SQLBulk-Operations()`関数を使用して、配置された`UPDATE`および`DELETE`操作を許可します。
Bytea as Long VarBinary	ByteaAsLong-VarBinary	Off	有効にすると、ドライバは BYTEA 値を`SQL_LONGVARIABLE`型であるかのように扱います。無効にすると、BYTEA 値は`SQL_VARIABLE`値として扱われます。
Bytea as LO	ByteaAsLO	False	有効にすると、ドライバは BYTEA 値をラージオブジェクトとして扱います。
Row versioning	RowVersioning	Off	`Row Versioning`オプションは、テーブル内の列をレポートするときにドライバが`xmin`列を含めるかどうかを指定します。`xmin`値は、行を作成したトランザクションの ID です。`SQL_CONCURRENCY = SQL_CONCUR_ROWVER`にカーソルを作成する場合は、行バージョン管理を使用する必要があります。
Disallow Premature	DisallowPremature	No/0	クエリを実行せずにクエリに関する情報を取得しようとした場合のドライバの動作を決定しクエリ。`Yes`の場合、ドライバはクエリのカーソルを宣言し、カーソルからメタデータを取得します。`No`の場合、ドライバはメタデータを要求するとすぐにコマンドを実行します。
True is -1	TrueIsMinus1	Off/0	<i>TrueIsMinus1</i> は、-1として真の BOOL 値を結果ようにドライバに指示します。このオプションが有効になっていない場合、ドライバは真の BOOL 値を`1`として返します。ドライバは、`0`として常に FALSE の BOOL 値を返します。

次のページに続く

表 1 – 前のページからの続き

Property	Keyword name	Default value	Description
Server side prepare	UseServerSidePre- pare	No/0	有効にすると、ドライバは`PRE-PARE`および`EXECUTE`コマンドを使用して、準備/実行モデルを実装します。
Use GSSAPI for GSS request	GssAuthUseGSS	False/0	`True/1`に設定されている場合、ドライバはGSSAPI 認証要求をサーバーに送信します。 Windows のみ。
Int8 As	BI	0	`BI`の値は、ドライバが`BIG-INT`の値を処理する方法を決定します。 8 `SQL_DOUBLE`として、 4 として `SQL_INTEGER`として、 12 として `SQL_VARCHAR`として、 0 として (MS Jet クライアント上)、`SQL_NUMERIC`として、 他のクライアントで0の場合、`SQL_BIGINT`として。

次のページに続く

表 1 – 前のページからの続き

Property	Keyword name	Default value	Description
Extra options />>Connect Settings	AB />ConnSettings	0x0	0x1-短い形式の接続文字列の出力を強制します。MFC CDatabase クラスを使用している場合、このオプションを指定します。0x2-MS Access が PostgreSQL のシリアルタイプをオートナンバータイプとして認識できるようにします。0x4-の ANSI 文字タイプを返します。アプリケーションからの問い合わせ。Unicode データのハンドリングが困難なアプリケーションには、このオプションを指定します。0x8-設定すると、NULL 日付は空の文字列として報告され、空の文字列は入力時に NULL 日付として解釈されます。0x10-`SQLGetInfo` がすべてのテーブルに関する情報を返すか、アクセス可能なテーブルのみを返すかを決定します。設定すると、アクセス可能なテーブルについてのみ情報が返されます。0x20-設定すると、各 SQL コマンドは個別のネットワークラウンドトリップで処理されます。それ以外の場合、SQL コマンドは、ネットワークレイテンシを減らすために、可能な限り少ないラウンドトリップにグループ化されます。ドライバがサーバーに接続するときに実行される SQL コマンドのセミコロン区切りリストが含まれます。
	Socket	4096	ドライバがクライアントへの接続に使用するバッファサイズを指定します。
	Lie	Off	有効にすると、ドライバはサポートされていない ODBC 機能をサポートすると主張します。
Lowercase Identifier	LowerCaseIdentifier	Off	有効にすると、ドライバは識別子を小文字に変換します。

次のページに続く

表 1 – 前のページからの続き

Property	Keyword name	Default value	Description
Disable Genetic Optimizer	Optimizer	Yes/1	遺伝的クエリーオプティマイザを無効にします。
Allow Keyset	UpdatableCursors	Yes/1	キーセットドリブンカーソルを許可する
SSL mode	SSLMode	Disabled	libpq（およびその依存関係）が EDB- ODBC ドライバと同じディレクトリにインストールされている場合、SSL モードを有効にすると、SSL およびその他のユーティリティを使用できます。
Force Abbreviated Connection String	CX	No/0	接続文字列の省略形を強制するオプションを有効にします。
Fake MSS	FakeOidIndex	No/0	MS SQL Server を偽装し、MS Access が PostgreSQL のシリアルタイプをオートナンバー型として認識できるようにします。
BDE Environment	BDE	No/0	このオプションを有効にすると、EDB- ODBC がチューニングされ、Borland Database Engine 準拠の出力（Unicode に関連）に対応します。
XA_Opt	INI_XAOPT	Yes/1	有効にすると、`SQL_TABLES` への呼び出しには、ユーザーがアクセス可能なテーブルのみが含まれます。

5.1 Windows でのデータソース定義の追加

Windows ODBC データソースアドミニストレーターは、名前付けデータソースを作成するグラフィカルインタフェースです。ODBC データソースアドミニストレーターを開くには、コントロールパネルで**管理ツール**メニューを開き、適切な ODBC データソースアイコン（32 ビットまたは 64 ビット）をダブルクリックします。

[**追加**] を選択して、[新しいデータソースの作成] ダイアログボックスを開きます。ドライバーのリストから **EnterpriseDB (ANSI)** または **EnterpriseDB (UNICODE)** を選択し、**Finish** を選択します。

[EnterpriseDB ODBC Driver] ダイアログボックスのフィールドを使用して、名前付けデータソースを定義します。

- [**データベース**] フィールドにデータベース名前を入力します。
- [**サーバー**] フィールドに EDB Postgres Advanced Server のホスト名前または IP アドレスを入力します。
- [**ユーザー名**] フィールドにユーザの名前を入力します。

- [説明] フィールドに名前付けデータソースのわかりやすい名前を入力します。libpq が EDB- ODBC ドライバと同じディレクトリにインストールされている場合、**SSL モード** label の横のリストがアクティブになり、SSL および他の EDB Postgres Advanced Server ユーティリティを使用できます。
- デフォルトポート番号 (5444) を受け入れるか、[ポート] フィールドに別の番号を入力します。
- [パスワード] フィールドにユーザのパスワードを入力します。

[オプション] ボックスにある [データソース] を選択して、[詳細オプション] ダイアログボックスを開き、接続プロパティを指定します。

[グローバル] を選択して、EDB- ODBC ドライバ (データソースではなく、ドライバ) のロギングオプションを指定できるダイアログを開きます。

- 遺伝的クエリーオプティマイザを無効にするには、[遺伝的オプティマイザを無効にする] を選択します。デフォルトでは、クエリーオプティマイザはオンになっています。
- **KSQO (キーセットクエリ最適化)** を選択して、キーセットクエリのサーバサイドサポートを有効にします。デフォルトでは、**キーセットクエリの最適化** はオンになっています。
- [一意のインデックスを認識する] を選択して、SQLStatistics() ファンクションが一意性インデックスをレポートするようにします。このオプションが選択されていない場合、SQLStatistics() ファンクションは、すべてのインデックスが重複値を許可することを報告します。デフォルトでは、**一意のインデックスを認識する** はオンになっています。
- [宣言/フェッチを使用] を選択して、アプリケーションが SELECT コマンドを実行するたびにドライバがサーバサイドカーソルを使用するように指定します。デフォルトでは、**UseDeclare/Fetch** はオフです。
- **CommLog (C : :raw-latex:ログ)** を選択して、すべてのクライアント/サーバートラフィックをログファイルに記録します。デフォルトでは、ロギングはオフです。
- **ステートメントの解析** を選択して、ドライバではなく (サーバーではなく)、SQLNumResultCols()、SQLDescribeCol()、または SQLColAttributes() ファンクションを呼び出すときに、シンプル SELECT ステートメントを解析ように指定します。デフォルトでは、このオプションはオフです。
- [**FreeStmt** としてキャンセル (Exp)] を選択して、SQLCancel() ファンクションがユーザーに代わって SQLFreeStmt(SQLClose) を呼び出すように指定します。デフォルトでは、このオプションはオフです。
- **MyLog (C : :raw-latex:ログ)** を選択して、ドライバアクティビティの詳細なレコードをログファイルに記録します。ログファイルの名前付けは c:\mylog\ \ *process-id*.log です。デフォルトでは、ロギングはオフです。

[不明なサイズ] ボックスのラジオボタンは、SQLDescribeCol() および SQLColAttributes() 関数が不明な型の列のサイズを計算する方法を指定します。

- **Maximum** を選択して、ドライバが VARCHAR または LONGVARCHAR (**UnknownsasLongVarChar** 設定に依存) で許可される最大サイズをレポートするように指定します。UnknownsasLongVarChar が有効になっている場合、ドライバは LONGVARCHAR の最大サイズ (**Miscellaneous** ボックスの **MaxLongVarChar** フィールドで指定) を返します。UnknownsasLongVarChar がクリアされている場合、ドライバは **Miscellaneous** ボックスの **MaxVarChar** フィールドで指定されたサイズを返します。
- ****不明*** を選択して、ドライバが **不明** の長さをレポートするように指定します。

- [**最長**] を選択して、ドライバが結果セットを検索し、見つかった最長値をレポートするように指定します。**UseDeclareFetch** が有効になっている場合は、**Longest** を指定しないでください。

[**データタイプオプション**] ボックスのプロパティは、ドライバが特定のタイプの列を処理する方法を決定します。

- **Text** を **LongVarChar** として選択すると、TEXT 値が SQL_LONGVARCHAR 型であるかのように処理されます。オフにすると、ドライバは TEXT 値を SQL_VARCHAR 値として扱います。デフォルトでは、TEXT 値は SQL_LONGVARCHAR 値として扱われます。
- **UnknownsasLongVarChar** を選択して、不明なタイプの値を SQL_LONGVARCHAR 値として扱うようにドライバに指定します。クリアすると、ドライバは不明なタイプの値を SQL_VARCHAR 値として扱います。デフォルトでは、不明なタイプの値は SQL_VARCHAR 値として扱われます。
- **BoolsasChar** を選択して、ドライバが BOOL 値を SQL_CHAR 値として扱うように指定します。クリアすると、BOOL 値は SQL_BIT 値として扱われます。デフォルトでは、BOOL 値は SQL_CHAR 値として扱われます。

[**その他**] ボックスのフィールドで、名前付けデータソースに関連付けられたいくつかのプロパティの値を指定できます。

- **MaxVarChar** フィールドに VARCHAR 値に許可される最大長さを示します。デフォルトでは、この値は 255 に設定されています。
- **MaxLongVarChar** フィールドに LONGVARCHAR 値に許可される最大長さを入力します。デフォルトでは、この値は 8190 に設定されています。
- **キャッシュサイズ** フィールドで、ドライバによってフェッチされる行数 (**UseDeclareFetch** が有効な場合) を指定します。デフォルト値は 100 です。
- **SysTablePrefixes** フィールドを使用して、テーブルがシステムテーブルであることを示すプレフィックスのセミコロン区切りリストを指定します。デフォルトでは、リストには **dd_;** が含まれています。

[**デフォルト**] を選択すると、このダイアログボックスの値をデフォルト設定にリセットできます。

[**適用**] ボタンを選択して、データソースプロパティへの変更を適用します。[**OK**] を選択して変更を適用し、終了します。

[**詳細オプション**] ダイアログボックスの左上隅にある [**ページ 2**] を選択して、2 番目の詳細オプションセットにアクセスします。

- **読み取り専用** を選択して、ドライバが次のコマンドを実行しないようにします：INSERT、UPDATE、DELETE、CREATE、ALTER、DROP、GRANT、REVOKE または LOCK。読み取り専用オプションを呼び出すと、ODBC のプロシージャコールエスケープ構文 (call=procedure-name?) を使用する呼び出しも防止されます。デフォルトでは、このオプションはオフです。
- **ShowSystemTables** を選択して、SQLTables() ファンクションの結果セットにシステムテーブルを含めます。このオプションが有効な場合、ドライバには、名前で始まるテーブル、または [**詳細オプション**] ダイアログボックスの [**ページ 1**] の [**SysTablePrefixes**] フィールドにリストされているプレフィックスのいずれかが含まれます。デフォルトでは、このオプションはオフです。
- **Showsys/dboTables[Access]** を選択して、ODBC データソースを介して sys スキーマおよび dbo スキーマ内のオブジェクトにアクセスします。デフォルトでは、このオプションはオンです。

- **Showsys/SQLTables** を選択して、MS SQL を使用してリンクサーバーのデータベーステーブルにアクセスできるようにします。
- **SQLRowCount()** メソッドの呼び出しが実行されたときに、**INSERT** ステートメントのパラメータ設定の配列全体に対して単一の累積行カウントが返されるようにするには、**挿入の累積行カウント**を選択します。このオプションをオフにすると、配列内の各パラメータ設定で個別の行カウントが使用できるため、**SQLRowCount()** の呼び出しは最後に挿入され行のカウントを返します。
- **LF<->CR/LF** 変換を選択して、サーバーから文字値をフェッチするときに改行文字をキャリッジリターン/改行フィードのペアに変換し、次の場合に改行文字に戻すようにドライバに指示します。サーバーに文字値を送信します。デフォルトでは、このオプションはオンです。
- **更新可能なカーソル** を選択して、ドライバが **SQLSetPos()** または **SQLBulkOperations()** 関数を使用して位置付けされた **UPDATE** および **DELETE** 操作を許可するように指定します。デフォルトでは、このオプションはオンです。
- **bytea** を **LO** として選択して、**BYTEA** の値を **SQL_LONGVARBINARY** の値として扱うようにドライバに指定します。オフにすると、EDB- ODBC は **BYTEA** 値を **SQL_VARBINARY** 値であるかのように扱います。デフォルトでは、**BYTEA** 値は **SQL_VARBINARY** 値として扱われます。
- **[行のバージョン管理]** を選択して、テーブルの列をレポートするときに **xmin** 列を含めます。**xmin** 列は、行を作成したトランザクションの ID です。**SQL_CONCURRENCY = SQL_CONCUR_ROWVER** の場所にカーソルを作成する場合は、行バージョン管理を使用する必要があります。デフォルトでは、このオプションはオフです。
- **早すぎるを禁止** を選択して、ドライバが実際にクエリーを実行せずにクエリーに関するメタデータ（結果セットの列数または列タイプ）を取得するように指定します。このオプションがオフの場合、クエリーに関するメタデータを要求すると、ドライバはクエリーを実行します。デフォルトでは、このオプションはオフです。
- **Trueis-1** を選択して、**True** の **BOOL** 値を **-1** として結果ようにドライバに指示します。このオプションがオフの場合、ドライバは **True** の **BOOL** 値を **1** として返します。ドライバは、**False** の **BOOL** 値を **0** として常に返します。
- **サーバー側準備** を選択して、**PREPARE** および **EXECUTE** コマンドを使用して **Prepare/Execute** モデルを実装するようドライバに指示します。デフォルトでは、このオプションはオンです。
- **[GSS リクエストに gssapi を使用]** を選択して、サーバーに **GSSAPI** 接続リクエストを送信するようドライバに指示します。
- **dbms_name** フィールドにデータベースシステム（**EnterpriseDB** または **PostgreSQL**）を入力します。ここに入力された値は、**SQLGetInfo()** ファンクションが呼び出されたときに **SQL_DBMS_NAME** 引数に返されます。デフォルトは **EnterpriseDB** です。

[Int8As] ボックスのオプションを使用して、ドライバがクライアントに **BIGINT** 値を返す方法を指定します。クライアントが **MS Jet** の場合、**デフォルト** を選択して **NUMERIC** のデフォルトタイプを指定します。クライアントが他の **ODBC** クライアントの場合、**BIGINT** を選択します。オプションで、ドライバが **BIGINT** 値を **bigint** (**SQL_BIGINT**)、**numeric** (**SQL_NUMERIC**)、**varchar** (**SQL_VARCHAR**)、**double** (**SQL_DOUBLE**)、または **int4** (**SQL_INTEGER**) として結果ように指定できます。

ExtraOpts フィールドのデフォルト値は **0x0** です。 **ExtraOpts** では、表に示されているオプションを指定できます。

Option	Specifies
0x1	Forces the output of short-length formatted connection string. Select this option when you're using the MFC CDatabase class.
0x2	Allows MS Access to recognize PostgreSQL's serial type as AutoNumber type.
0x4	Returns ANSI character types for the inquiries from applications. Select this option for applications that have difficulty handling Unicode data.
0x8	If set, NULL dates are reported as empty strings, and empty strings are interpreted as NULL dates on input.
0x10	Determines if <i>SQLGetInfo</i> returns information about all tables or only accessible tables. If set, information is returned only for accessible tables.
0x20	If set, each SQL command is processed in a separate network round trip. Otherwise, SQL commands are grouped into as few round trips as possible to reduce network latency.

Protocol ボックスには、特定のフロントエンド/バックエンドプロトコルバージョンを使用してサーバーと対話するようにドライバに指示するオプションが含まれています。デフォルトでは、選択されるプロトコルは **7.4+** です。オプションで、バージョン **6.4+**、**6.3**、または **6.2** から選択できます。

[エラー時のロールバックのレベル**] ボックスには、ドライバがエラーハンドリングを処理する方法を指定するオプションが含まれています。

Option	Specifies
Transaction	If the driver encounters an error, it rolls back the current transaction.
Statement	If the driver encounters an error, it rolls back the current statement.
Nop	If the driver encounters an error, you must manually roll back the current transaction before the application can continue.

[**OID オプション**] ボックスには、一部のテーブルに含まれる OID 列をドライバが公開する方法を制御するオプションが含まれています。

- [**ShowColumn**] を選択して、*SQLColumns()* ファンクションの結果セットに OID 列を含めます。クリアすると、OID 列は *SQLColumns()* から非表示になります。
- [**FakeColumns**] を選択して、*SQLStatistics()* ファンクションで各 OID 列に一意インデックスが存在することをレポートするように指定します。

[**接続設定**] フィールドを使用して、この接続を開くときに使用するドライバのパラメータ割り当てのリストを指定します。SET ステートメントで変更できる設定パラメータは、セミコロンで区切られたリストに含めることができます。例：

```
set search_path to company1,public;
```

名前付けデータソースの接続プロパティを定義した後、**適用**を選択してオプションを適用します。[**OK**] を選択してオプションを保存し、終了します。

[EnterpriseDB ODBC Driver] ダイアログボックスで [**Global**] を選択して、[Global Settings] ダイアログボックスを開きます。このダイアログボックスのオプションは、EDB- ODBC ドライバのロギングオプションを制御します。このダイアログボックスを使用して、名前付けデータソースなしでドライバを使用する

場合、または接続文字列が解析される前に発生するドライバ操作のロギングをロギングするように強制します。

- **CommLog** を選択して、すべてのクライアント/サーバートラフィックをログファイルに記録します。ログファイルの名前付けは `C:\psqlodbc_<process-id>` で、`<process-id>` は使用中のプロセスの名前です。
- **[マイログ]** を選択して、ドライバーのアクティビティのログファイルを保持します。ログファイルの名前付けは `c:\mylog_<process-id>` です。 `<process-id>` は使用中のプロセスの名前です。
- **[ロギングのフォルダ]** フィールドでログファイルの場所を指定します。名前付けデータソースの接続情報を入力したら、**テスト** を選択して、ドライバマネージャが定義済みのデータソースに接続できることを確認します。

[OK] を選択して、**[接続テスト]** ダイアログボックスを終了します。接続が成功した場合は、**保存** を選択して名前付けデータソースを保存します。接続の確立に問題がある場合は、パラメータを調整して再度テストします。

5.2 Linux でのデータソース定義の追加

Linux ODBC アドミニストレータは、unixODBC と共に配布されるグラフィカルツール。ODBC アドミニストレータを使用して、ODBC ドライバと名前付けリソースを管理できます。システムに ODBC アドミニストレータを追加するには、ターミナルウィンドウを開き、スーパーユーザ権限を引き受けて、次のように入力します。

```
yum install unixODBC
```

```
yum install unixODBC-kde
```

ODBC アドミニストレータを呼び出すには、ターミナルウィンドウを開き、**ODBCConfig** と入力します。

EDB Postgres Advanced Server Connectors コンポーネントをインストールすると、ODBC Administrator のドライバーのリストに EDB- ODBC ドライバが追加されます。**[詳細]** を選択し、**[ドライバー]** タブを選択して、**enterprisedb** ドライバがリストに表示されることを確認します。

EDB- ODBC ドライバがドライバーのリストに表示されない場合は、ODBC Administrator を使用して追加できます。ドライバ定義を追加するには、**[ドライバー]** タブを選択し、**[追加]** を選択します。

[ドライバーのプロパティ] ウィンドウで、EDB- ODBC ドライバをドライバマネージャに登録します。

- **Name** フィールドにドライバの一意の名前を追加します。
- **[説明]** フィールドにドライバの説明を追加します。

EDB の場所へのパスを追加します- **ドライバー** フィールドに ODBC ドライバ。デフォルトでは、ドライバへの完全なパスは `/usr/edb/odbc/lib/edb-odbc.so` です。

EDB の場所へのパスを追加します- **セットアップ** フィールドに ODBC ドライバセットアップファイル。デフォルトでは、ドライバセットアップファイルへの完全なパスは `/usr/edb/odbc/lib/libodbcedbS.so` です。

EDB- ODBC ドライバのドライバプロパティを入力したら、**OK** を選択します。[ODBC データソースアドミニストレーター] ウィンドウの EDB- ODBC ドライバが、利用可能な ODBC ドライバーのリストに含まれるようになりました。

ドライバマネージャで利用可能な EDB- ODBC ドライバを使用すると、データソースを追加できます。左側のパネルで [**データソース名**] オプションを選択し、追加するデータソース名前のタイプに適した DSN タブを選択します。

- [**ユーザー**] タブを選択して、現在のユーザのみが使用できる名前付けデータソースを追加します。データソースは `/user/.odbc.ini` に格納されます。
- [**システム**] タブを選択して、すべてのユーザーが使用できる名前付けデータソースを追加します。すべてのシステムデータソースは、通常 `/etc/odbc.ini` という 1 つのファイルに保存されます。
- [**ファイル**] タブを選択して、すべてのユーザーが使用できるが、選択したファイルに保存されている名前付けデータソースを追加します。

適切なタブを選択し、**追加** を選択します。[新しいデータソースの作成] ウィンドウが開きます。

リストから EDB- ODBC ドライバを選択し、**OK** を選択して、データソースプロパティウィンドウを開きます。

EDB- ODBC ドライバの接続プロパティを指定して、**データソースプロパティ (新規)** ウィンドウに入力します。

- [**名前**] フィールドにデータソース名前を入力します。
- [**説明**] フィールドに名前付けデータソースの説明を入力します。
- unixODBC ドライバには、ODBC アプリケーションからログファイルへのシーケンスの呼び出しを記録するトレースユーティリティが含まれています。**Trace** フィールドに **Yes** を指定して、トレースユーティリティをオンにします。トレースユーティリティを使用すると、アプリケーションの速度が低下する可能性があります。
- **TraceFile** フィールドを使用して、Trace ユーティリティによって返される情報を受信するファイルを指定します。
- [**データベース**] フィールドに EDB Postgres Advanced Server データベースの名前を入力します。
- [**サーバー名**] フィールドに EDB Postgres Advanced Server のホスト名前または IP アドレスを入力します。
- [**ユーザー名**] フィールドにユーザの名前を入力します。
- [**パスワード**] フィールドにユーザのパスワードを入力します。
- [**ポート**] フィールドにポート番号を入力します（またはデフォルト値の 5444 を受け入れます）。
- **Protocol** フィールドを使用して、フロントエンド/バックエンドプロトコルバージョンを指定します。デフォルト値は **7.4** です。オプションで、プロトコルバージョン **7.4**、**6.4**、**6.3**、**6.2** から選択できます。
- **Yes** を指定するには、**ReadOnly** フィールドを使用して、ドライバが INSERT、UPDATE、DELETE、CREATE、ALTER、DROP、GRANT、REVOKE または LOCK のコマンドを実行できないようにします。**読み取り専用** オプションを有効にすると、ODBC プロシージャコールエスケープ構文（`call=procedure-name?`）を使用する呼び出しも防止されます。デフォルトでは、**ReadOnly** は **No** に設定されています。

- テーブルの列をレポートするときにドライバに **xmin** 列を含める場合は、**RowVersioning** フィールドを使用して **Yes** を指定します。xmin 列は、行を作成したトランザクションの ID です。SQL_CONCURRENCY = SQL_CONCUR_ROWVER にカーソルを作成する場合は、行のバージョン管理を使用する必要があります。デフォルトでは、**RowVersioning** は **No** に設定されています。
- **SQLTables()** ファンクションの結果セットにシステムテーブルを含める場合は、**ドライバ** フィールドを使用して **Yes** を指定します。デフォルトでは、このフィールドは **いいえ** に設定されています。
- **SQLColumns()** ファンクションの結果セットにドライバ列を含める場合は、**ShowOidColumn** フィールドを使用して **Yes** を指定します。**ShowOidColumn** が **No** に設定されている場合、OID 列は **SQLColumns()** から非表示になります。デフォルトでは、このオプションは **いいえ** に設定されています。
- **SQLStatistics()** ファンクションで各 OID 列に一意インデックスが存在することをレポートする場合は、**FakeOidIndex** フィールドを使用して **Yes** を指定します。これは、アプリケーションに一意の識別子がニーズで、テーブルに識別子が含まれていない場合に便利です。デフォルト値は **いいえ** です。
- **ConnSettings** フィールドを使用して、この接続を開くときに使用するドライバのパラメータ割り当てのリストを指定します。

接続プロパティを定義したら、**OK** を選択します。新しいデータソースがデータソース名のリストに追加されます。

第6章 Driver functionality

ODBC 関数を使用して、EDB- ODBC とサーバー間の接続のさまざまな属性に関する特定の情報を ODBC にクエリーできます。

- `SQLGetInfo()` は、EDB- ODBC ドライバと EDB Postgres Advanced Server に関する情報を返します。
- `SQLGetEnvAttr()` は、ODBC 環境属性に関する情報を返します。
- `SQLGetConnectAttr()` は、個々の接続に固有の属性に関する情報を返します。
- `SQLGetStmtAttr()` は、個々のステートメントに固有の属性に関する情報を返します。

ODBC 関数を使用して、ODBC とのインタフェースに使用するオブジェクトの属性を設定することもできます。

- `SQLSetConnectAttr()` ファンクションを使用して、接続属性を設定します。
- `SQLSetEnvAttr()` ファンクションを使用して、環境属性を設定します。
- `SQLSetStmtAttr()` ファンクションを使用して、ステートメントの属性を設定します。

6.1 SQLGetInfo ()

ODBC `SQLGetInfo()` ファンクションは、EDB- ODBC ドライバと EDB Postgres Advanced Server に関する情報を返します。 `info_type` として `SQL_ODBC_VER` を指定しない限り、`SQLGetInfo()` を呼び出すためのオープン接続が必要です。 `SQLGetInfo()` のシグネチャは次のとおりです。

```
SQLRETURN SQLGetInfo
(
    SQLHDBC conn_handle , // Input
    SQLUSMALLINT info_type , // Input
    SQLPOINTER info_pointer , // Output
    SQLSMALLINT buffer_len , // Input
    SQLSMALLINT * string_length_pointer // Output
);
```

- `conn_handle` & mdash; 接続ハンドル。
- `info_type` & mdash; `SQLGetInfo ()` が取得している情報のタイプ。
- `info_pointer` & mdash; 取得した値を保持するメモリバッファへのポインタ。

`info_type` 引数が `SQL_DRIVER_HDESC` または `SQL_DRIVER_HSTMT` の場合、`info_pointer` 引数は Input と Output の両方です。

- `buffer_len` & mdash; `info_pointer` が指す、割り当てられたメモリバッファの長さ。 `info_pointer` が NULL の場合、 `buffer_len` は無視されます。 戻り値が固定サイズの場合、 `buffer_len` は無視されます。 `buffer_len` は、要求された値が文字の並びのフォームで返される場合にのみ使用されます。
- `string_length_pointer` & mdash; `SQLSMALLINT` 値へのポインタ。 `SQLGetInfo()` は、要求された値のサイズをこの整数に書き込みます。

典型的な使用法は、 NULL `info_pointer` を指定して `SQLGetInfo()` を呼び出して要求された値の長さを取得し、必要なバイト数を割り当ててから、 `SQLGetInfo()` を再度呼び出して（新しく割り当てられたバッファのアドレスを提供して）実際の値を取得します。最初の呼び出しは、値を保持するために必要なバイト数を取得します。2 番目の呼び出しは値を取得します。

戻り値のサイズが `buffer_len` を超える場合、情報は切り捨てられ、 NULL は終了します。戻り値が固定サイズの場合、 `string_length` は無視され、要求された値のサイズは `SQLGetInfo()` によって提供されません。

`SQLGetInfo()` は、次の形式のいずれかで情報を書き込みます。

- `SQLINTEGER` ビットマスク
- `SQLINTEGER` フラグ
- `SQLINTEGER` バイナリ値
- `SQLSMALLINT` 値
- NULL で終わる文字の並び

`SQLGetInfo()` は、 `SQL_SUCCESS`、 `SQL_SUCCESS_WITH_INFO`、 `SQL_ERROR`、または `SQL_INVALID_HANDLE` を返します。

次の表に、 EDB Postgres Advanced Server 接続に関する EDB- ODBC によって返される情報を示します。

SQL info_type argument and description**	EDB_ODBC/EDB Postgres Advanced Server returns
<code>SQL_ACCESSIBLE_PROCEDURES</code> : Indicates if procedures returned by <i>SQLProcedures()</i> can be executed by the application.	Returns <i>N</i> . Some procedures executed by the <i>SQLProcedures()</i> function might be executed by the application.
<code>SQL_ACCESSIBLE_TABLES</code> : Indicates if the user has SELECT privileges on all table names returned by <i>SQLTables()</i> .	Returns <i>N</i> . The user might not have select privileges on one or more tables returned by the <i>SQLTables()</i> function.
<code>SQL_ACTIVE_CONNECTIONS</code> prev. <code>SQL_MAX_DRIVER_CONNECTIONS</code> : Indicates the maximum number of connections EDB-ODBC can support.	Returns 0. There's no specified limit to the number of connections allowed.
<code>SQL_ACTIVE_ENVIRONMENTS</code> : The number of active environments EDB-ODBC can support.	Returns 0. There's no specified limit to the number of environments allowed.
<code>SQL_ACTIVE_STATEMENTS</code> prev. <code>SQL_MAX_CONCURRENT_ACTIVITIES</code> : Indicates the maximum number of active statements EDB-ODBC can support.	Returns 0. There's no specified limit to the number of active statements allowed.
<code>SQLAggregateFunction</code> : Identifies the aggregate functions supported by the server and driver.	Returns <i>SQL_AF_ALL</i> .

次のページに続く

表 1 – 前のページからの続き

SQL info_type argument and description**	EDB_ODBC/EDB Postgres Advanced Server returns
SQL_ALTER_DOMAIN: Identifies the <i>ALTER DOMAIN</i> clauses supported by the server.	Returns 0. <i>ALTER DOMAIN</i> clauses aren't supported.
SQL_ALTER_TABLE: Identifies the <i>ALTER TABLE</i> clauses supported by the server.	Returns <i>SQL_AT_ADD_COLUMN</i> , <i>SQL_AT_DROP_TABLE_CONSTRAINT_CASCADE</i> , <i>SQL_AT_DROP_TABLE_CONSTRAINT</i> , <i>SQL_AT_CONSTRAINT_INITIALLY_DEFERRED</i> , <i>SQL_AT_CONSTRAINT_INITIALLY_IMMEDIATE</i> , <i>SQL_AT_CONSTRAINT_DEFERRABLE</i>
SQL_ASYNC_MODE: Level of asynchronous mode supported by EDB-ODBC.	Returns <i>SQL_AM_NONE</i> . Asynchronous mode isn't supported.
SQL_BATCH_ROW_COUNT: Indicates how the driver returns row counts.	Returns <i>SQL_BRC_EXPLICIT</i> . Row counts are available when executed by calling <i>SQLExecute</i> or <i>SQLExecDirect</i> .
SQL_BATCH_SUPPORT: Indicates support for batch statement execution.	Returns <i>SQL_BS_SELECT_EXPLICIT</i> , <i>SQL_BS_ROW_COUNT_EXPLICIT</i> . The driver supports explicit batches with result set and row count generating statements.
SQL_BOOKMARK_PERSISTENCE: Indicates level of support for bookmarks.	Returns <i>SQL_BP_DELETE</i> , <i>SQL_BP_TRANSACTION</i> , <i>SQL_BP_UPDATE</i> , <i>SQL_BP_SCROLL</i> .
SQL_CATALOG_LOCATION Now SQL_QUALIFIER_LOCATION: Indicates the position of the catalog in a qualified table name.	Returns <i>SQL_CL_START</i> . The catalog portion of a qualified table name is at the beginning of the name.
SQL_CATALOG_NAME Now SQL_QUALIFIER_NAME: Indicates support for catalog names.	Returns Y. The server supports catalog names.
SQL_CATALOG_NAME_SEPARATOR Now SQL_QUALIFIER_NAME_SEPARATOR: Character separating the catalog name from the adjacent name element.	Returns ' '. The server expects a ' ' character between the qualifier and the table name.
SQL_CATALOG_TERM Now SQL_QUALIFIER_TERM: The term used to describe a catalog.	Returns catalog.
SQL_CATALOG_USAGE Now SQL_QUALIFIER_USAGE: Indicates the SQL statements that can refer to catalogs.	Returns <i>SQL_CU_DML_STATEMENTS</i> . Catalog names can be used in <i>SELECT</i> , <i>INSERT</i> , <i>UPDATE</i> , <i>DELETE</i> , <i>SELECT FOR UPDATE</i> and positioned <i>UPDATE</i> and <i>DELETE</i> statements.
SQL_COLLATION_SEQ: Returns the name of the collation sequence.	Returns an empty string. The name of the default collation is unknown.
SQL_COLUMN_ALIAS: Indicates server support for column aliases.	Returns Y. The server supports column aliases.

次のページに続く

表 1 – 前のページからの続き

SQL info_type argument and description**	EDB_ODBC/EDB Postgres Advanced Server returns
SQL_CONCAT_NULL_BEHAVIOR: Indicates how the server handles concatenation of NULL values.	Returns <i>SQL_CB_NON_NULL</i> . Concatenating a NULL value and a non-NULL value results in a NULL value.
SQL_CONVERT_BIGINT: Indicates conversion support from the <i>BIGINT</i> type using the <i>CONVERT</i> function.	Returns 0. The server doesn't support conversion.
SQL_CONVERT_BINARY: Indicates conversion support from the <i>BINARY</i> type using the <i>CONVERT</i> function.	Returns 0. The server doesn't support conversion.
SQL_CONVERT_BIT: Indicates conversion support from the <i>BIT</i> type using the <i>CONVERT</i> function.	Returns <i>SQL_CVT_INTEGER</i> , <i>SQL_CVT_BIT</i> .
SQL_CONVERT_CHAR: Indicates conversion support from the <i>CHAR</i> type using the <i>CONVERT</i> function.	Returns 0. The server doesn't support conversion.
SQL_CONVERT_DATE: Indicates conversion support from the <i>DATE</i> type using the <i>CONVERT</i> function.	Returns 0. The server doesn't support conversion.
SQL_CONVERT_DECIMAL: Indicates conversion support from the <i>DECIMAL</i> type using the <i>CONVERT</i> function.	Returns 0. The server doesn't support conversion.
SQL_CONVERT_DOUBLE: Indicates conversion support from the <i>DOUBLE</i> type using the <i>CONVERT</i> function.	Returns 0. The server doesn't support conversion.
SQL_CONVERT_FLOAT: Indicates conversion support from the <i>FLOAT</i> type using the <i>CONVERT</i> function.	Returns 0. The server doesn't support conversion.
SQL_CONVERT_FUNCTIONS: Lists the scalar conversion functions supported by the server and driver using the <i>CONVERT</i> function.	Returns <i>SQL_FN_CVT_CONVERT</i> .
SQL_CONVERT_INTEGER: Lists the conversion support from the <i>INTEGER</i> type using the <i>CONVERT</i> function.	Returns <i>SQL_CVT_INTEGER</i> , <i>SQL_CVT_BIT</i> .
SQL_CONVERT_INTERVAL_DAY_TIME: Indicates conversion support from the <i>INTERVAL_DAY_TIME</i> type using the <i>CONVERT</i> function.	This info_type isn't currently supported.
SQL_CONVERT_INTERVAL_YEAR_MONTH: Indicates conversion support from the <i>INTERVAL_YEAR_MONTH</i> type using the <i>CONVERT</i> function.	This info_type isn't currently supported.
SQL_CONVERT_LONGVARBINARY: Indicates conversion support for the <i>LONG_VARBINARY</i> type using the <i>CONVERT</i> function.	Returns 0. The server doesn't support conversion.
SQL_CONVERT_LONGVARCHAR: Indicates conversion support for the <i>LONGVARCHAR</i> type using the <i>CONVERT</i> function.	Returns 0. The server doesn't support conversion.
SQL_CONVERT_NUMERIC: Indicates conversion support for the <i>NUMERIC</i> type using the <i>CONVERT</i> function.	Returns 0. The server doesn't support conversion.
SQL_CONVERT_REAL: Indicates conversion support for the <i>REAL</i> type using the <i>CONVERT</i> function	Returns 0. The server doesn't support conversion.
SQL_CONVERT_SMALLINT: Indicates conversion support for the <i>SMALLINT</i> type using the <i>CONVERT</i> function.	Returns <i>SQL_CVT_INTEGER</i> , <i>SQL_CVT_BIT</i> .

次のページに続く

表 1 – 前のページからの続き

SQL info_type argument and description**	EDB_ODBC/EDB Postgres Advanced Server returns
SQL_CONVERT_TIME: Indicates conversion support for TIME type using the <i>CONVERT</i> function.	Returns 0. The server doesn't support conversion.
SQL_CVT_TIMESTAMP: Indicates conversion support for TIMESTAMP type using the <i>CONVERT</i> function.	Returns 0. The server doesn't support conversion.
SQL_CONVERT_TINYINT: Indicates conversion support for the TINYINT type using the <i>CONVERT</i> function.	Returns <i>SQL_CVT_INTEGER</i> , <i>SQL_CVT_BIT</i> .
SQL_CONVERT_VARBINARY: Indicates conversion support for the VARBINARY type using the <i>CONVERT</i> function.	Returns 0. The server doesn't support conversion.
SQL_CONVERT_VARCHAR: Indicates conversion support for VARCHAR type using the <i>CONVERT</i> function.	Returns <i>SQL_CVT_INTEGER</i> , <i>SQL_CVT_BIT</i> .
SQL_CONVERT_WCHAR: Indicates conversion support for the WCHAR type using the <i>CONVERT</i> function.	This info_type is valid only when using the Unicode driver. Returns 0. The server doesn't support conversion.
SQL_CONVERT_WLONGVARCHAR: Indicates conversion support for the WLONGVARCHAR type using the <i>CONVERT</i> function.	This info_type is valid only when using the Unicode driver. Returns 0. The server doesn't support conversion.
SQL_CONVERT_WVARCHAR: Indicates conversion support for the WVARCHAR type using the <i>CONVERT</i> function.	This info_type is valid only when using the Unicode driver. Returns 0. The server doesn't support conversion.
SQL_CORRELATION_NAME: Indicates server support for correlation names.	Returns <i>SQL_CN_ANY</i> . Correlation names are supported and can be any valid name.
SQL_CREATE_ASSERTION: Indicates support for the <i>CREATE ASSERTION</i> statement.	Returns 0. The <i>CREATE ASSERTION</i> statement isn't supported.
SQL_CREATE_CHARACTER_SET: Indicates support for the <i>CREATE CHARACTER</i> statement.	Returns 0. The <i>CREATE CHARACTER</i> statement isn't supported.
SQL_CREATE_COLLATION: Indicates support for the <i>CREATE COLLATION</i> statement.	Returns 0. The <i>CREATE COLLATION</i> statement isn't supported.
SQL_CREATE_DOMAIN: Indicates support for the <i>CREATE DOMAIN</i> statement.	Returns 0. The <i>CREATE DOMAIN</i> statement isn't supported.
SQL_CREATE_SCHEMA: Indicates support for the <i>CREATE SCHEMA</i> statement.	Returns <i>SQL_CS_CREATE_SCHEMA</i> , <i>SQL_CS_AUTHORIZATION</i> .
SQL_CREATE_TABLE: Indicates support for the <i>CREATE TABLE</i> statement.	Returns <i>SQL_CT_CREATE_TABLE</i> , <i>SQL_CT_GLOBAL_TEMPORARY</i> , <i>SQL_CT_CONSTRAINT_INITIALLY_DEFERRED</i> , <i>SQL_CT_CONSTRAINT_INITIALLY_IMMEDIATE</i> , <i>SQL_CT_CONSTRAINT_DEFERRABLE</i> , <i>SQL_CT_COLUMN_CONSTRAINT</i> , <i>SQL_CT_COLUMN_DEFAULT</i> , <i>SQL_CT_TABLE_CONSTRAINT</i> , <i>SQL_CT_CONSTRAINT_NAME_DEFINITION</i>

次のページに続く

表 1 – 前のページからの続き

SQL info_type argument and description**	EDB_ODBC/EDB Postgres Advanced Server returns
SQL_CREATE_TRANSLATION: Indicates support for the <i>CREATE TRANSLATION</i> statement.	Returns 0. The <i>CREATE TRANSLATION</i> statement isn't supported.
SQL_CREATE_VIEW: Indicates support for the <i>CREATE VIEW</i> statement.	Returns <i>SQL_CV_CREATE_VIEW</i> .
SQL_CURSOR_COMMIT_BEHAVIOR: Indicates how a <i>COMMIT</i> operation affects the cursor.	Returns <i>SQL_CB_PRESERVE</i> . Cursors are unchanged and can continue to fetch data.
SQL_CURSOR_ROLLBACK_BEHAVIOR: Indicates the server behavior after a <i>ROLLBACK</i> operation.	Returns <i>SQL_CB_PRESERVE</i> . Cursors are unchanged and can continue to fetch data.
SQL_CURSOR_SENSITIVITY: Indicates how the server synchronizes changes to a result set.	This info_type isn't currently supported.
SQL_DATA_SOURCE_NAME: Returns the server name used during connection.	The value returned is determined by the connection properties.
SQL_DATA_SOURCE_READ_ONLY: Indicates if the connection is in READ ONLY mode.	The value returned is determined by the connection properties.
SQL_DATABASE_NAME: Returns the name of the database.	The value returned is determined by the connection properties.
SQL_DATETIME_LITERALS: Indicates the DATETIME LITERALS supported by the server.	This info_type is not supported.
SQL_DBMS_NAME: Returns the name of the DBMS system.	Returns the value given by the <i>dbms_name</i> parameter from the <i>odbc.ini</i> file on Linux or the <i>dbms_name</i> field of page 2 of the Advanced Options dialog box when defining a data source in Windows. The default is <i>EnterpriseDB</i> .
SQL_DBMS_VER: Returns the server version.	Determined by the server.
SQL_DDL_INDEX: Indicates support for creating and dropping indexes.	Returns <i>SQL_DI_CREATE_INDEX</i> , <i>SQL_DI_DROP_INDEX</i> .
SQL_DEFAULT_TXN_ISOLATION: Indicates support for transaction isolation by the server.	Returns <i>TXN_READ_COMMITTED</i> . Nonrepeatable or phantom reads are possible. Dirty reads aren't.
SQL_DESCRIBE_PARAMETER: Indicates support for the <i>DESCRIBE INPUT</i> statement.	Returns <i>N</i> . The <i>DESCRIBE INPUT</i> statement isn't supported.
SQL_DM_VER: The version of the driver manager.	Determined by driver manager.
SQL_DRIVER_HDBC: The driver's connection handle.	Returns an <i>SQLULEN</i> value that contains the driver's connection handle.
SQL_DRIVER_HDESC: The driver descriptor handle.	Returns an <i>SQLULEN</i> value that contains driver's descriptor handle.
SQL_DRIVER_HENV: The driver's environment handle.	Returns an <i>SQLULEN</i> value that contains the driver's environment handle.

次のページに続く

表 1 – 前のページからの続き

SQL info_type argument and description**	EDB_ODBC/EDB Postgres Advanced Server returns
SQL_DRIVER_HLIB: The driver handle.	Returns an <i>SQLULEN</i> value that contains the library handle (returned to the ODBC driver manager when the manager loaded the driver).
SQL_DRIVER_HSTMT: The driver's statement handle.	Returns an <i>SQLULEN</i> value that contains the driver's statement handle.
SQL_DRIVER_NAME: The name of the driver.	Returns <i>EDB-ODBC.DLL</i>
SQL_DRIVER_ODBC_VER: Identifies the ODBC version that the driver supports.	Returns <i>03.50</i>
SQL_DRIVER_VER: Identifies the driver version.	Returns <i>9.0.0.6</i>
SQL_DROP_ASSERTION: Lists the <i>DROP ASSERTION</i> clauses supported by the server.	Returns <i>0</i> .
SQL_DROP_CHARACTER_SET: Lists the <i>DROP CHARACTER</i> clauses supported by the server.	Returns <i>0</i> .
SQL_DROP_COLLATION: Lists the <i>DROP COLLATION</i> clauses supported by the server.	Returns <i>0</i> .
SQL_DROP_DOMAIN: Lists the <i>DROP DOMAIN</i> clauses supported by the server.	Returns <i>0</i> .
SQL_DROP_SCHEMA: Lists the <i>DROP SCHEMA</i> clauses supported by the server.	Returns <i>SQL_DS_DROP_SCHEMA</i> , <i>SQL_DS_RESTRICT</i> , <i>SQL_DS_CASCADE</i> .
SQL_DROP_TABLE: Lists the <i>DROP TABLE</i> clauses supported by the server.	Returns <i>SQL_DT_DROP_TABLE</i> , <i>SQL_DS_RESTRICT</i> , <i>SQL_DS_CASCADE</i> .
SQL_DROP_TRANSLATION: Lists the <i>DROP TRANSLATION</i> clauses supported by the server.	Returns <i>0</i> .
SQL_DROP_VIEW: Lists the <i>DROP VIEW</i> clauses supported by the server.	Returns <i>SQL_DV_DROP_VIEW</i> , <i>SQL_DS_RESTRICT</i> , <i>SQL_DS_CASCADE</i> .
SQL_DYNAMIC_CURSOR_ATTRIBUTES1: Describes the first set of dynamic cursor attributes supported by the driver.	Returns <i>0</i> .
SQL_DYNAMIC_CURSOR_ATTRIBUTES2: Describes the second set of dynamic cursor attributes supported by the driver.	Returns <i>0</i> .
SQL_EXPRESSIONS_IN_ORDERBY: Indicates server support for <i>ORDER BY</i> .	Returns <i>Y</i> .
SQL_FETCH_DIRECTION: Indicates <i>FETCH</i> order options (deprecated in ODBC 3.0).	Returns <i>SQL_FD_FETCH_NEXT</i> , <i>SQL_FD_FETCH_FIRS</i> , <i>SQL_FD_FETCH_LAST</i> , <i>SQL_FD_FETCH_PRIOR</i> , <i>SQL_FD_FETCH_ABSOLUTE</i> , <i>SQL_FD_FETCH_RELATIVE</i> , <i>SQL_FD_FETCH_BOOKMARK</i> .

次のページに続く

表 1 – 前のページからの続き

SQL info_type argument and description**	EDB_ODBC/EDB Postgres Advanced Server returns
SQL_FILE_USAGE: Indicates how a single-tier driver treats files on the server.	Returns <i>SQL_FILE_NOT_SUPPORTED</i> . The driver isn't a single-tier file.
SQL_FORWARD_ONLY_CURSOR_ATTRIBUTES1: Describes the forward-only cursor attributes supported by the driver.	Returns <i>SQL_CA1_NEXT</i> .
SQL_FORWARD_ONLY_CURSOR_ATTRIBUTES2: Describes extended attributes for the forward-only cursor designated by <i>SQL_FORWARD_ONLY_CURSOR_ATTRIBUTES1</i> .	Returns <i>SQL_CA2_READ_ONLY_CONCURRENCY</i> , <i>SQL_CA2_CRC_EXACT</i> .
SQL_GETDATA_EXTENSIONS: Lists supported extensions to <i>SQLGetData</i> .	Returns <i>SQL_GD_ANY_COLUMN</i> , <i>SQL_GD_ANY_ORDER</i> , <i>SQL_GD_BLOCK</i> , <i>SQL_GD_BOUND</i> .
SQL_GROUP_BY: Indicates the relationship between a <i>GROUP BY</i> clause and columns in the <i>SELECT</i> list.	Returns <i>SQL_GB_GROUP_BY_EQUALS_SELECT</i> .
SQL_IDENTIFIER_CASE: Indicates case sensitivity and case storage of SQL identifiers.	Returns <i>SQL_IC_LOWER</i> .
SQL_INDEX_KEYWORDS: Indicates support for the <i>CREATE INDEX</i> statement.	Returns <i>SQL_IK_NONE</i> .
SQL_INFO_SCHEMA_VIEWS: Lists the views supported in the <i>INFORMATION_SCHEMA</i> .	Returns 0.
SQL_INTEGRITY Prev. SQL_ODBC_SQL_OPT_IEF: Indicates server support for referential integrity syntax checking.	Returns N.
SQL_INSERT_STATEMENT: Indicates level of support for the <i>INSERT</i> statement.	Returns <i>SQL_IS_INSERT_LITERALS</i> , <i>SQL_IS_INSERT_SEARCHED</i> , <i>SQL_IS_SELECT INTO</i> .
SQL_KEYSET_CURSOR_ATTRIBUTES1: Describes the first set of keyset cursor attributes supported by the driver.	Returns <i>SQL_CA1_NEXT</i> , <i>SQL_CA1_ABSOLUTE</i> , <i>SQL_CA1_RELATIVE</i> , <i>SQL_CA1_BOOKMARK</i> , <i>SQL_CA1_LOCK_NO_CHANGE</i> , <i>SQL_CA1_POS_POSITION</i> , <i>SQL_CA1_POS_UPDATE</i> , <i>SQL_CA1_POS_DELETE</i> , <i>SQL_CA1_POS_REFRESH</i> , <i>SQL_CA1_BULK_ADD</i> , <i>SQL_CA1_BULK_UPDATE_BY_BOOKMARK</i> , <i>SQL_CA1_BULK_DELETE_BY_BOOKMARK</i> , <i>SQL_CA1_BULK_FETCH_BY_BOOKMARK</i> .

次のページに続く

表 1 – 前のページからの続き

SQL info_type argument and description**	EDB_ODBC/EDB Postgres Advanced Server returns
SQL_KEYSET_CURSOR_ATTRIBUTES2: Describes the second set of keyset cursor attributes supported by the driver.	Returns <i>SQL_CA2_READ_ONLY_CONCURRENCY</i> , <i>SQL_CA2_OPT_ROWVER_CONCURRENCY</i> , <i>SQL_CA2_SENSITIVITY_ADDITIONS</i> , <i>SQL_CA2_SENSITIVITY_DELETIONS</i> , <i>SQL_CA2_SENSITIVITY_UPDATES</i> , <i>SQL_CA2_CRC_EXACT</i> .
SQL_KEYWORDS: Identifies the server-specific reserved keywords.	Returns “”. There are no server-specific reserved keywords.
SQL_LIKE_ESCAPE_CLAUSE: Indicates support for an escape character in <i>LIKE</i> predicates.	Returns <i>N</i> . EDB Postgres Advanced Server doesn't support escape characters in <i>LIKE</i> predicates.
SQL_LOCK_TYPES: Lists supported lock types (deprecated in ODBC 3.0).	Returns <i>SQL_LCK_NO_CHANGE</i> .
SQL_MAX_ASYNC_CONCURRENT_STATEMENTS: The number of active concurrent statements that the driver can support.	This info_type is currently unsupported.
SQL_MAX_BINARY_LITERAL_LEN: The maximum length of a binary literal.	Returns <i>0</i> . The maximum length is unspecified.
SQL_MAX_CATALOG_NAME_LEN: The maximum length of a catalog name on the server.	Returns <i>0</i> . The maximum length is unspecified.
SQL_MAX_QUALIFIER_NAME_LEN: The maximum length of a qualifier.	Returns <i>0</i> . The maximum length is unspecified.
SQL_MAX_CHAR_LITERAL_LEN: The maximum number of characters in a character string.	Returns <i>0</i> . The maximum length is unspecified.
SQL_MAX_COLUMN_NAME_LEN: The maximum length of a column name.	Returns <i>64</i> . Column names can't exceed 64 characters in length.
SQL_MAX_COLUMNS_IN_GROUP_BY: The maximum number of columns allowed in a <i>GROUP BY</i> clause.	Returns <i>0</i> . The maximum length is unspecified.
SQL_MAX_COLUMNS_IN_INDEX: The maximum number of columns allowed in an index.	Returns <i>0</i> . The maximum length is unspecified.
SQL_MAX_COLUMNS_IN_ORDER_BY: The maximum number of columns allowed in an <i>ORDER BY</i> clause.	Returns <i>0</i> . The maximum length is unspecified.
SQL_MAX_COLUMNS_IN_SELECT: The maximum number of columns allowed in a <i>SELECT</i> list.	Returns <i>0</i> . The maximum length is unspecified.
SQL_MAX_COLUMNS_IN_TABLE: The maximum number of columns allowed in a table.	Returns <i>0</i> . The maximum length is unspecified.
SQL_MAX_CONCURRENT_ACTIVITIES prev. SQL_MAX_ACTIVE_STATEMENTS: The maximum number of active SQL statements that the driver can support.	Returns <i>0</i> . The maximum length is unspecified.
SQL_MAX_CURSOR_NAME_LEN: The maximum length of a cursor name.	Returns <i>32</i> . A cursor name can't exceed 32 characters in length.

次のページに続く

表 1 – 前のページからの続き

SQL info_type argument and description**	EDB_ODBC/EDB Postgres Advanced Server returns
SQL_MAX_DRIVER_CONNECTIONS prev. SQL_ACTIVE_CONNECTIONS: The maximum number of active connections the driver can support.	Returns 0. There is no specified limit to the number of connections supported.
SQL_MAX_IDENTIFIER_LEN: The maximum identifier length allowed by the server.	Returns 64. Identifiers can't exceed 64 characters in length.
SQL_MAX_INDEX_SIZE: The maximum number of bytes allowed in the (combined) fields of an index.	Returns 0. The maximum size is unspecified.
SQL_MAX_OWNER_NAME_LEN Now SQL_MAX_SCHEMA_NAME_LEN: The maximum length of an owner name allowed by the server.	Returns 64. The maximum length of an owner name is 64 characters.
SQL_MAX_PROCEDURE_NAME_LEN: The maximum length of a procedure name allowed by the server.	Returns 0. The maximum length is unspecified.
SQL_MAX_QUALIFIER_NAME_LEN Now SQL_MAX_CATALOG_NAME_LEN: The maximum length of a qualifier name allowed by the server.	Returns 0. The maximum length of a qualifier is unspecified.
SQL_MAX_ROW_SIZE: The maximum length of a row.	Returns 0. The maximum row length is unspecified.
SQL_MAX_ROW_SIZE_INCLUDES_LONG: Indicates whether the <i>SQL_MAX_ROW_SIZE</i> includes the length of any <i>LONGVARCHAR</i> or <i>LONGVARBINARY</i> columns in the row.	Returns Y. <i>SQL_MAX_ROW_SIZE</i> includes the length of any <i>LONGVARCHAR</i> or <i>LONGVARBINARY</i> columns in the row.
SQL_MAX_SCHEMA_NAME_LEN: The maximum length of a schema name allowed by the server.	Returns 64. The maximum length of a schema name is 64 characters.
SQL_MAX_STATEMENT_LEN: The maximum length of a SQL statement.	Returns 0. Maximum statement length is limited by available memory.
SQL_MAX_TABLE_NAME_LEN: The maximum length of a table name allowed by the server.	Returns 64. The maximum length of a table name is 64 characters.
SQL_MAX_TABLES_IN_SELECT: The maximum number of tables allowed in the <i>FROM</i> clause of a <i>SELECT</i> statement.	Returns 0. The maximum number of tables allowed is unspecified.
SQL_MAX_USER_NAME_LEN: The maximum length of the user name allowed by the server.	Returns 0. The maximum length of a user name is unspecified.
SQL_MULT_RESULT_SETS: Indicates server support for multiple result sets.	Returns Y. EDB Postgres Advanced Server supports multiple result sets.
SQL_MULTIPLE_ACTIVE_TXN: Indicates if the server supports multiple active transactions.	Returns Y. EDB Postgres Advanced Server supports multiple active transactions.
SQL_NEED_LONG_DATA_LEN: Indicates if the server needs the length of a <i>LONG</i> data value before receiving the value.	Returns N. EDB Postgres Advanced Server doesn't need the length of a <i>LONG</i> data value before receiving the value.
SQL_NON_NULLABLE_COLUMNS: Indicates if the server supports NOT NULL values in columns.	Returns <i>SQL_NNC_NON_NULL</i> . EDB Postgres Advanced Server does support NOT NULL values in columns.

次のページに続く

表 1 – 前のページからの続き

SQL info_type argument and description**	EDB_ODBC/EDB Postgres Advanced Server returns
SQL_NULL_COLLATION: Indicates where NULL values are located in a result set.	Returns <i>SQL_NC_HIGH</i> . The location of NULL values in a data set is determined by the <i>ASC</i> and <i>DESC</i> keywords. NULL values are sorted to the high end of the data set.
SQL_NUMERIC_FUNCTIONS: Lists the numeric functions supported by the driver and the server.	Returns <i>SQL_FN_NUM_ABS</i> , <i>SQL_FN_NUM_ATAN</i> , <i>SQL_FN_NUM_CEILING</i> , <i>SQL_FN_NUM_COS</i> , <i>SQL_FN_NUM_EXP</i> , <i>SQL_FN_NUM_FLOOR</i> , <i>SQL_FN_NUM_LOG</i> , <i>SQL_FN_NUM_MOD</i> , <i>SQL_FN_NUM_SIGN</i> , <i>SQL_FN_NUM_SIN</i> , <i>SQL_FN_NUM_SQRT</i> , <i>SQL_FN_NUM_TAN</i> , <i>SQL_FN_NUM_RAND</i> , <i>SQL_FN_NUM_POWER</i> , <i>SQL_FN_NUM_ROUND</i> .
SQL_ODBC_API_CONFORMANCE: Indicates the ODBC 3.0 compliance level.	Returns <i>SQL_OAC_LEVEL1</i> . The driver conforms to ODBC Level 1 interface.
SQL_ODBC_INTERFACE_CONFORMANCE: Indicates the ODBC interface that the driver adheres to.	Returns <i>SQL_OIC_CORE</i> .
SQL_ODBC_SAG_CLI_CONFORMANCE: Indicates the SQL Access Group compliance level that the driver adheres to.	Returns <i>SQL_OSCC_NOT_COMPLIANT</i> . The driver isn't SAG CLI compliant.
SQL_ODBC_SQL_CONFORMANCE: Indicates the SQL grammar level that the driver conforms to.	Returns <i>SQL_OSC_CORE</i> . The driver conforms to the core grammar level.
SQL_ODBC_SQL_OPT_IEF Now SQL_INTEGRITY: Indicates server support for referential integrity syntax checking.	Returns <i>N</i> . The server doesn't support referential integrity syntax checking.
SQL_ODBC_VER: The ODBC version supported by the driver manager	Returns <i>03.52.0000</i> .
SQL_OJ_CAPABILITIES: Identifies the outer joins that are supported by the server.	Returns <i>SQL_OJ_LEFT</i> , <i>SQL_OJ_RIGHT</i> , <i>SQL_OJ_FULL</i> , <i>SQL_OJ_NESTED</i> , <i>SQL_OJ_NOT_ORDERED</i> , <i>SQL_OJ_INNER</i> , <i>SQL_OJ_ALL_COMPARISON_OPS</i> .
SQL_OUTER_JOINS: Indicates support for outer joins and the outer join escape sequence.	Returns <i>Y</i> . Outer joins are supported.
SQL_OWNER_TERM prev. SQL_SCHEMA_TERM: The term used to describe a schema.	Returns schema.

次のページに続く

表 1 – 前のページからの続き

SQL info_type argument and description**	EDB_ODBC/EDB Postgres Advanced Server returns
SQL_ORDER_BY_COLUMNS_IN_SELECT: Indicates if the columns in an <i>ORDER BY</i> clause must be included in the <i>SELECT</i> list.	Returns <i>N</i> . Columns in an <i>ORDER BY</i> clause don't have to be in the <i>SELECT</i> list.
SQL_OWNER_USAGE prev. SQL_SCHEMA_USAGE: Returns a string that indicates which statements support schema qualifiers.	Returns <i>SQL_OU_DML_STATEMENTS</i> , <i>SQL_OU_TABLE_DEFINITION</i> , <i>SQL_OU_INDEX_DEFINITION</i> , <i>SQL_OU_PRIVILEGE_DEFINITION</i> .
SQL_PARAM_ARRAY_ROW_COUNTS: Indicates if the server returns a single row count or separate row counts for each element in an array when executing a parameterized statement with at least one parameter bound to the array.	Returns <i>SQL_PARC_BATCH</i> if separate row counts are available for each element in an array. Returns <i>SQL_PARC_NO_BATCH</i> if a single, cumulative row count is available for the entire array.
SQL_PARAM_ARRAY_SELECTS: Indicates if the server returns one result set or a separate result set for each element in an array (or if the driver doesn't allow this feature) when executing a parameterized statement with at least one parameter bound to the array.	Returns <i>SQL_PAS_BATCH</i> . One data set is available for each element in an array.
SQL_POS_OPERATION: Lists the options supported by <i>SQLSetPos()</i> .	Returns <i>SQL_POS_POSITION</i> , <i>SQL_POS_REFRESH</i> , <i>SQL_POS_UPDATE</i> , <i>SQL_POS_DELETE</i> , <i>SQL_POS_ADD</i> .
SQL_POSITIONED_STATEMENTS: Lists the supported positioned SQL statements.	Returns <i>SQL_PS_POSITIONED_DELETE</i> , <i>SQL_PS_POSITIONED_UPDATE</i> , <i>SQL_PS_SELECT_FOR_UPDATE</i> .
SQL_PROCEDURE_TERM: The term used to describe a procedure.	Returns procedure.
SQL_PROCEDURES: Indicates if the server and the driver support SQL procedures and procedure invocation syntax.	Returns <i>Y</i> . The server and driver support procedures and procedure invocation syntax.
SQL_QUALIFIER_LOCATION prev. SQL_CATALOG_LOCATION: Indicates the position of the schema name in a qualified table name.	Returns <i>SQL_CL_START</i> . The catalog portion of a qualified table name is at the beginning of the name.
SQL_QUALIFIER_NAME prev. SQL_CATALOG_NAME: Indicates server support for catalog names.	Returns <i>Y</i> . The server supports catalog names.
SQL_QUALIFIER_NAME_SEPARATOR prev. SQL_CATALOG_NAME_SEPARATOR: Character separating the qualifier name from the adjacent name element.	Returns <i>'.'</i> . The server expects a <i>'.'</i> character between the qualifier and the table name.
SQL_QUALIFIER_TERM prev. SQL_CATALOG_TERM: The term used to describe a qualifier.	Returns catalog.

次のページに続く

表 1 – 前のページからの続き

SQL info_type argument and description**	EDB_ODBC/EDB Postgres Advanced Server returns
SQL_QUALIFIER_USAGE prev. SQL_CATALOG_USAGE: Indicates the SQL statements that might refer to qualifiers.	Returns <i>SQL_CU_DML_STATEMENTS</i> . Catalog names can be used in <i>SELECT</i> , <i>INSERT</i> , <i>UPDATE</i> , <i>DELETE</i> , <i>SELECT FOR UPDATE</i> , and positioned <i>UPDATE</i> and <i>DELETE</i> statements.
SQL_QUALIFIER_USAGE Now SQL_CATALOG_USAGE: Identifies DML statements that support qualifier names.	Returns <i>SQL_CU_DML_STATEMENTS</i> . Qualifiers can be used in all DML statements (<i>SELECT</i> , <i>INSERT</i> , <i>UPDATE</i> , <i>DELETE</i> , <i>SELECT FOR UPDATE</i>).
SQL_QUOTED_IDENTIFIER_CASE: Indicates case sensitivity of quoted identifiers.	Returns <i>SQL_IC_SENSITIVE</i> . Quoted identifiers are case sensitive.
SQL_CATALOG_NAME_SEPARATOR prev. SQL_QUALIFIER_NAME_SEPARATOR: The character that separates the name qualifier from the name element.	Returns '.'. The '.' character is used as a separator in qualified names.
SQL_QUALIFIER_TERM: The term used to describe a qualifier.	Returns catalog
SQL_QUALIFIER_LOCATION: The position of the qualifier in a qualified table name.	Returns <i>SQL_CL_START</i> . The qualifier precedes the table name in a qualified table name.
SQL_ROW_UPDATES: Indicates if keyset-driven or mixed cursors maintain row versions or values.	Returns <i>Y</i> . Cursors maintain values for all fetched rows and can detect updates to the row values.
SQL_SCHEMA_TERM: The term used to describe a schema.	Returns schema.
SQL_SCHEMA_USAGE: Indicates the SQL statements that might refer to schemas.	Returns <i>SQL_OU_DML_STATEMENTS</i> , <i>SQL_OU_TABLE_DEFINITION</i> , <i>SQL_OU_INDEX_DEFINITION</i> , <i>SQL_OU_PRIVILEGE_DEFINITION</i> .
SQL_SCROLL_CONCURRENCY: Indicates the cursor concurrency control options supported by the server.	Returns <i>SQL_SCCO_READ_ONLY</i> , <i>SQL_SCCO_OPT_ROWVER</i> .
SQL_SCROLL_OPTIONS: Indicates the cursor scroll options supported by the server.	Returns <i>SQL_SO_FORWARD_ONLY</i> , <i>SQL_SO_KEYSET_DRIVEN</i> , <i>SQL_SO_STATIC</i> .
SQL_SEARCH_PATTERN_ESCAPE: The escape character that allows use of the wildcard characters % and _ in search patterns.	Returns ". The " character is used as an escape character for the '%' and '_' characters in search patterns.
SQL_SERVER_NAME: Indicates the name of the host.	The returned value is determined by connection properties.
SQL_SPECIAL_CHARACTERS: Indicates any special characters allowed in identifier names.	Returns _. The underscore character is allowed in identifier names.
SQL_SQL_CONFORMANCE: Indicates the level of SQL-92 compliance.	Returns <i>SQL_SC_SQL92_ENTRY</i> . The driver is SQL92 entry-level compliant.

次のページに続く

表 1 – 前のページからの続き

SQL info_type argument and description**	EDB_ODBC/EDB Postgres Advanced Server returns
SQL_SQL92_DATETIME_FUNCTIONS: Lists the datetime functions supported by the server.	Returns <i>SQL_SDF_CURRENT_DATE</i> , <i>SQL_SDF_CURRENT_TIME</i> , <i>SQL_SDF_CURRENT_TIMESTAMP</i> .
SQL_SQL92_FOREIGN_KEY_DELETE_RULE: Indicates the server-enforced rules for using a foreign key in a <i>DELETE</i> statement.	Returns <i>SQL_SFKD_CASCADE</i> , <i>SQL_SFKD_NO_ACTION</i> , <i>SQL_SFKD_SET_DEFAULT</i> , <i>SQL_SFKD_SET_NULL</i> .
SQL_SQL92_FOREIGN_KEY_UPDATE_RULE: Indicates the server-enforced rules for using a foreign key in an <i>UPDATE</i> statement.	Returns <i>SQL_SFKU_CASCADE</i> , <i>SQL_SFKU_NO_ACTION</i> , <i>SQL_SFKU_SET_DEFAULT</i> , <i>SQL_SFKU_SET_NULL</i> .
SQL_SQL92_GRANT: Indicates the supported <i>GRANT</i> statement clauses.	Returns <i>SQL_SG_DELETE_TABLE</i> , <i>SQL_SG_INSERT_TABLE</i> , <i>SQL_SG_REFERENCES_TABLE</i> , <i>SQL_SG_SELECT_TABLE</i> , <i>SQL_SG_UPDATE_TABLE</i> .
SQL_SQL92_NUMERIC_VALUE_FUNCTIONS: Lists the scalar numeric functions supported by the server and driver.	Returns <i>SQL_SNVF_BIT_LENGTH</i> , <i>SQL_SNVF_CHAR_LENGTH</i> , <i>SQL_SNVF_CHARACTER_LENGTH</i> , <i>SQL_SNVF_EXTRACT</i> , <i>SQL_SNVF_OCTET_LENGTH</i> , <i>SQL_SNVF_POSITION</i> .
SQL_SQL92_PREDICATES, Identifies the predicates of a <i>SELECT</i> statement supported by the server.	Returns <i>SQL_SP_EXISTS</i> , <i>SQL_SP_ISNOTNULL</i> , <i>SQL_SP_ISNULL</i> , <i>SQL_SP_OVERLAPS</i> , <i>SQL_SP_LIKE</i> , <i>SQL_SP_IN</i> , <i>SQL_SP_BETWEEN</i> , <i>SQL_SP_COMPARISON</i> , <i>SQL_SP_QUANTIFIED_COMPARISON</i> .
SQL_SQL92_RELATIONAL_JOIN_OPERATORS: Identifies the relational join operators supported by the server.	Returns <i>SQL_SRJO_CROSS_JOIN</i> , <i>SQL_SRJO_EXCEPT_JOIN</i> , <i>SQL_SRJO_FULL_OUTER_JOIN</i> , <i>SQL_SRJO_INNER_JOIN</i> , <i>SQL_SRJO_INTERSECT_JOIN</i> , <i>SQL_SRJO_LEFT_OUTER_JOIN</i> , <i>SQL_SRJO_NATURAL_JOIN</i> , <i>SQL_SRJO_RIGHT_OUTER_JOIN</i> , <i>SQL_SRJO_UNION_JOIN</i> .

次のページに続く

表 1 – 前のページからの続き

SQL info_type argument and description**	EDB_ODBC/EDB Postgres Advanced Server returns
SQL_SQL92_REVOKE: Identifies the clauses in a <i>REVOKE</i> statement that are supported by the server.	Returns <i>SQL_SR_DELETE_TABLE</i> , <i>SQL_SR_INSERT_TABLE</i> , <i>SQL_SR_REFERENCES_TABLE</i> , <i>SQL_SR_SELECT_TABLE</i> , <i>SQL_SR_UPDATE_TABLE</i> .
SQL_SQL92_ROW_VALUE_CONSTRUCTOR: Indicates the row value constructor expressions in a <i>SELECT</i> statement that are supported by the server.	Returns <i>SQL_SRVC_VALUE_EXPRESSION</i> , <i>SQL_SRVC_NULL</i> .
SQL_SQL92_STRING_FUNCTIONS: Lists the string scalar functions supported by the server and driver.	Returns <i>SQL_SSF_CONVERT</i> , <i>SQL_SSF_LOWER</i> , <i>SQL_SSF_UPPER</i> , <i>SQL_SSF_SUBSTRING</i> , <i>SQL_SSF_TRANSLATE</i> , <i>SQL_SSF_TRIM_BOTH</i> , <i>SQL_SSF_TRIM_LEADING</i> , <i>SQL_SSF_TRIM_TRAILING</i> .
SQL_SQL92_VALUE_EXPRESSIONS: Indicates the value expressions supported by the server.	Returns <i>SQL_SVE_CASE</i> , <i>SQL_SVE_CAST</i> , <i>SQL_SVE_COALESCE</i> , <i>SQL_SVE_NULLIF</i> .
SQL_STANDARD_CLI_CONFORMANCE: Indicates the CLI standard the driver conforms to.	This info_type is currently unsupported.
SQL_STATIC_CURSOR_ATTRIBUTES1: Describes the first set of static cursor attributes supported by the driver.	Returns <i>SQL_CA1_NEXT</i> , <i>SQL_CA1_ABSOLUTE</i> , <i>SQL_CA1_RELATIVE</i> , <i>SQL_CA1_BOOKMARK</i> , <i>SQL_CA1_LOCK_NO_CHANGE</i> , <i>SQL_CA1_POS_POSITION</i> , <i>SQL_CA1_POS_UPDATE</i> , <i>SQL_CA1_POS_DELETE</i> , <i>SQL_CA1_POS_REFRESH</i> , <i>SQL_CA1_BULK_ADD</i> , <i>SQL_CA1_BULK_UPDATE_BY_BOOKMARK</i> , <i>SQL_CA1_BULK_DELETE_BY_BOOKMARK</i> , <i>SQL_CA1_BULK_FETCH_BY_BOOKMARK</i> .
SQL_STATIC_CURSOR_ATTRIBUTES2: Describes the second set of static cursor attributes supported by the driver.	Returns <i>SQL_CA2_READ_ONLY_CONCURRENCY</i> , <i>SQL_CA2_OPT_ROWVER_CONCURRENCY</i> , <i>SQL_CA2_SENSITIVITY_ADDITIONS</i> , <i>SQL_CA2_SENSITIVITY_DELETIONS</i> , <i>SQL_CA2_SENSITIVITY_UPDATES</i> , <i>SQL_CA2_CRC_EXACT</i> .

次のページに続く

表 1 – 前のページからの続き

SQL info_type argument and description**	EDB_ODBC/EDB Postgres Advanced Server returns
SQL_STATIC_SENSITIVITY: Indicates whether changes made to a static cursor by <i>SQLSetPos()</i> or <i>UPDATE</i> or <i>DELETE</i> statements are detected by the application.	Returns <i>SQL_SS_ADDITIONS</i> , <i>SQL_SS_DELETIONS</i> , <i>SQL_SS_UPDATES</i> .
SQL_STRING_FUNCTIONS: Lists the scalar string functions supported by the server and driver.	Returns <i>SQL_FN_STR_CONCAT</i> , <i>SQL_FN_STR_LTRIM</i> , <i>SQL_FN_STR_LENGTH</i> , <i>SQL_FN_STR_LOCATE</i> , <i>SQL_FN_STR_LCASE</i> , <i>SQL_FN_STR_RTRIM</i> , <i>SQL_FN_STR_SUBSTRING</i> , <i>SQL_FN_STR_UCASE</i> .
SQL_SUBQUERIES: Identifies the subquery predicates to a <i>SELECT</i> statement supported by the server.	Returns <i>SQL_SQ_COMPARISON</i> , <i>SQL_SQ_EXISTS</i> , <i>SQL_SQ_IN</i> , <i>SQL_SQ_QUANTIFIED</i> .
SQL_SYSTEM_FUNCTIONS: Lists the scalar system functions supported by the server and driver.	Returns 0.
SQL_TABLE_TERM: The term used to describe a table.	Returns table.
SQL_TIMEDATE_ADD_INTERVALS: Indicates the timestamp intervals supported by the server for the <i>TIMESTAMPADD</i> scalar function.	Returns 0.
SQL_TIMEDATE_DIFF_INTERVALS: Indicates the timestamp intervals supported by the server for the <i>TIMESTAMPDIFF</i> scalar function.	Returns 0.
SQL_TIMEDATE_FUNCTIONS: Indicates the date and time functions supported by the server.	Returns <i>SQL_FN_TD_NOW</i> , <i>SQL_FN_TD_CURDATE</i> , <i>SQL_FN_TD_CURTIME</i> .
SQL_TXN_CAPABLE: Identifies the transaction support offered by the server and driver.	Returns <i>SQL_TC_ALL</i> . Transactions can contain both DML and DDL statements.
SQL_TXN_ISOLATION_OPTION: Indicates the transaction isolation level supported by the server.	Returns <i>SQL_TXN_READ_COMMITTED</i> , <i>SQL_TXN_SERIALIZABLE</i> .
SQL_UNION: Indicates server support for the <i>UNION</i> clause.	Returns <i>SQL_U_UNION</i> , <i>SQL_U_UNION_ALL</i> .
SQL_USER_NAME: Identifies the name of the user connected to a database. Can be different than the login name.	This value is determined by the connection properties.
SQL_XOPEN_CLI_YEAR: The publication year of the X/Open specification that the driver manager complies with.	This info_type is currently unsupported.

6.2 接続属性

ODBC SQLGetConnectAttr() および SQLSetConnectAttr() 関数を使用して、接続属性の値を取得または設定できます。

6.2.1 SQLGetConnectAttr ()

SQLGetConnectAttr() ファンクションは、接続属性の現在の値を返します。シグネチャは次のとおりです。

```
SQLRETURN SQLGetConnectAttr
(
    SQLHDBC conn_handle, //Input
    SQLINTEGER attribute, //Input
    SQLPOINTER value_pointer, //Output
    SQLINTEGER buffer_length, //Input
    SQLINTEGER * string_length_pointer //Output
);
```

- `conn_handle` & mdash; 接続ハンドル。
- `attribute` & mdash; 値を取得する属性を識別します。
- `value_pointer` & mdash; `attribute` 値を受け取るメモリの場所へのポインタ。
- `buffer_length` & mdash; `attribute` が ODBC によって定義され、`value_pointer` が文字の並びまたはバイナリバッファを指す場合、`buffer_length` は `value_pointer` の長さです。`value_pointer` が固定サイズの値（整数など）を指している場合、`buffer_length` は無視されます。

EDB- ODBC が属性を定義する場合、SQLGetConnectAttr() は `buffer_length` パラメータを設定します。`buffer_length` は次のいずれかです。

Value type	Meaning
Character string	The length of the character string
Binary buffer	The result of <i>SQL_LEN_BINARY_ATTR(length)</i>
Fixed length data type	<i>SQL_IS_INTEGER</i> or <i>SQL_IS_UIINTEGER</i>
Any other type	<i>SQL_IS_POINTER</i>

- `string_length_pointer` & mdash; `value_pointer` で結果バイト数を受け取る SQLINTEGER へのポインタ。`value_pointer` が NULL の場合、`string_length_pointer` は返されません。

このファンクションは、SQL_SUCCESS、SQL_SUCCESS_WITH_INFO、SQL_NO_DATA、SQL_ERROR、または SQL_INVALID_HANDLE を返します。

次の表に、EDB- ODBC でサポートされる接続属性を示します。

Attribute	Supported?	Notes
SQL_ATTR_ACCESS_MODE	No	SQL_MODE_READ_WRITE

次のページに続く

表 3 – 前のページからの続き

Attribute	Supported?	Notes
SQL_ATTR_ASYNC_ENABLE	No	SQL_ASYNC_ENABLE_OFF
SQL_ATTR_AUTO_IPD	No	
SQL_ATTR_AUTOCOMMIT	Yes	SQL_AUTOCOMMIT, SQL_AUTOCOMMIT_ON, SQL_AUTOCOMMIT_OFF
SQL_ATTR_CONNECTION_TIMEOUT	No	
SQL_ATTR_CURRENT_CATALOG	No	
SQL_ATTR_DISCONNECT_BEHAVIOR	No	
SQL_ATTR_ENLIST_IN_DTC	Yes	For win32 and with conditional compilation
SQL_ATTR_ENLIST_IN_XA	No	
SQL_ATTR_LOGIN_TIMEOUT	No	SQL_LOGIN_TIMEOUT
SQL_ATTR_ODBC_CURSORS	No	
SQL_ATTR_PACKET_SIZE	No	
SQL_ATTR_QUIET_MODE	No	
SQL_ATTR_TRACE	No	
SQL_ATTR_TRACEFILE	No	
SQL_ATTR_TRANSLATE_LIB	No	
SQL_ATTR_TRANSLATE_OPTION	No	
SQL_ATTR_TXN_ISOLATION	Yes	SQL_TXN_ISOLATION, SQL_DEFAULT_TXN_ISOLATION

6.2.2 SQLSetConnectAttr ()

ODBC SQLSetConnectAttr() ファンクションを使用して、接続属性の値を設定できます。ファンクションのシグネチャは次のとおりです。

```
SQLRETURN SQLSetConnectAttr
(
    SQLHDBC conn_handle , // Input
    SQLINTEGER attribute , // Input
    SQLPOINTER value_pointer , // Input
    SQLINTEGER string_length , // Input
);
```

- `conn_handle` & mdash; 接続ハンドル。
- `attribute` & mdash; 値を設定する属性を識別します。
- `value_pointer` & mdash; 属性が想定する値へのポインタ。
- `string_length` & mdash; `attribute` が ODBC によって定義され、`value_pointer` がバイナリバッファまたは文字の並びを指す場合、`string_length` は `value_pointer` の長さです。 `value_pointer` が固定長の値（整数など）を指している場合、`string_length` は無視されます。

EDB- ODBC が属性を定義する場合、アプリケーションは `string_length` パラメータを設定します。可能な `string_length` 値を表に示します。

Value type	Meaning
Character string	The length of the character string or <i>SQL_NTS</i>
Binary buffer	The result of <i>SQL_LEN_BINARY_ATTR(length)</i>
Fixed length data type	<i>SQL_IS_INTEGER</i> or <i>SQL_IS_UINTEGER</i>
Any other type	<i>SQL_IS_POINTER</i>

`SQLSetConnectAttr()` は、`SQL_SUCCESS`、`SQL_SUCCESS_WITH_INFO`、`SQL_ERROR`、`SQL_STILL_EXECUTING` または `SQL_INVALID_HANDLE` を返します。

接続ハンドルが割り当てられた後はいつでも、`SQLFreeHandle()` の呼び出しで接続が閉じられるまで、`SQLSetConnectAttr()` を呼び出すことができます。呼び出しによって設定されたすべての属性は、`SQLFreeHandle()` の呼び出しまで保持されます。

接続属性には、設定可能な特定の時間枠があります。一部の属性は接続が確立される前に設定する必要がありますが、他の属性は接続が確立された後にのみ設定できます。

次の表は、接続属性とそれらを設定できる時間フレームを示しています。

Attribute	Set before or after establishing a connection?
<code>SQL_ATTR_ACCESS_MODE</code>	Before or after
<code>SQL_ATTR_ASYNC_ENABLE</code>	Before or after
<code>SQL_ATTR_AUTO_IPD</code>	Before or after
<code>SQL_ATTR_AUTOCOMMIT</code>	Before or after
<code>SQL_ATTR_CONNECTION_TIMEOUT</code>	Before or after
<code>SQL_ATTR_CURRENT_CATALOG</code>	Before or after
<code>SQL_ATTR_ENLIST_IN_DTC</code>	After
<code>SQL_ATTR_ENLIST_IN_XA</code>	After
<code>SQL_ATTR_LOGIN_TIMEOUT</code>	Before
<code>SQL_ATTR_ODBC_CURSORS</code>	Before
<code>SQL_ATTR_PACKET_SIZE</code>	Before
<code>SQL_ATTR_QUIET_MODE</code>	Before or after
<code>SQL_ATTR_TRACE</code>	Before or after
<code>SQL_ATTR_TRACEFILE</code>	Before or after
<code>SQL_ATTR_TRANSLATE_LIB</code>	After
<code>SQL_ATTR_TRANSLATE_OPTION</code>	After
<code>SQL_ATTR_TXN_ISOLATION</code>	Before or after

6.3 環境属性

ODBC SQLGetEnvAttr() および SQLSetEnvAttr() 関数を使用して、環境属性の値を取得または設定できます。

6.3.1 SQLGetEnvAttr ()

SQLGetEnvAttr() ファンクションを使用して、システムの環境属性の現在の値を見つけます。ファンクションのシグネチャは次のとおりです。

```
SQLRETURN SQLGetConnectAttr
(
    SQLHDBC env_handle, // Input
    SQLINTEGER attribute, // Input
    SQLPOINTER value_ptr, // Output
    SQLINTEGER buffer_length, // Input
    SQLINTEGER * string_length_pointer // Output
);
```

- `env_handle` & mdash; 環境ハンドル。
- `attribute` & mdash; 値を取得する属性を識別します。
- `value_pointer` & mdash; `attribute` 値を受け取るメモリの場所へのポインタ。
- `buffer_length` & mdash; 属性が文字の並びの場合、`buffer_length` は `value_ptr` の長さです。属性の値が文字の並びでない場合、`buffer_length` は使用されません。
- `string_length_pointer` & mdash; `value_pointer` で結果バイト数を受け取る `SQLINTEGER` へのポインタ。`value_pointer` が `NULL` の場合、`string_length_pointer` は返されません。

このファンクションは、`SQL_SUCCESS`、`SQL_SUCCESS_WITH_INFO`、`SQL_NO_DATA`、`SQL_ERROR`、または `SQL_INVALID_HANDLE` を返します。

次の表に、EDB- ODBC でサポートされる環境属性を示します。

Attribute	Supported?	Restrictions?
SQL_ATTR_CONNECTION_POOLING	SQL_CP_ONE_PER_DRIVER or SQL_CP_OFF	Determined by connection properties
SQL_ATTR_ODBC_VERSION	(SQL_OV_ODBC3), (SQL_OV_ODBC2)	None
SQL_ATTR_OUTPUT_NTS	SQL_SUCCESS	None

6.4 SQLSetEnvAttr ()

SQLSetEnvAttr() ファンクションを使用して、環境属性の値を設定できます。ファンクションのシグネチャは次のとおりです。

```
SQLRETURN SQLSetEnvAttr
(
    SQLHENV  env_handle , //Input
    SQLINTEGER attribute , //Input
    SQLPOINTER value_pointer , //Input
    SQLINTEGER string_length //Input
);
```

- `env_handle` & mdash; 環境ハンドル。
- `attribute` & mdash; 値を設定する属性を識別します。
- `value_pointer` & mdash; 属性に割り当てられた値へのポインタ。値は、指定された属性に応じて、NULL で終了文字の並びまたは 32 ビット整数値です。
- `string_length` & mdash; `value_pointer` がバイナリバッファまたは文字の並びへのポインタの場合、`string_length` は `value_pointer` の長さです。属性に割り当てられる値が文字の場合、`string_length` はその文字の並びの長さです。`value_pointer` が NULL の場合、`string_length` は返されません。`value_pointer` が整数の場合、`string_length` は無視されます。

SQLSetEnvAttr() は、SQL_SUCCESS 、 SQL_INVALID_HANDLE 、SQL_ERROR 、または SQL_SUCCESS_WITH_INFO を返します。アプリケーションは、接続ハンドルを割り当てる前に SQLSetEnvAttr() を呼び出す必要があります。環境属性に適用されるすべての値は、接続に対して SQLFreeHandle() が呼び出されるまで持続します。ODBC バージョン 3.x では、マルチプルの環境ハンドルを同時に割り当てることができます。

次の表に、SQLSetAttr() で設定できる環境属性を示します。

Attribute	Value_pointer type	Restrictions?
SQL_ATTR_ODBC_VERSION	32-bit integer	Set this attribute before the application calls any function that includes an <i>SQLHENV</i> argument.
SQL_ATTR_OUTPUT_NTS	32-bit integer	Defaults to <i>SQL_TRUE</i> . Calls that set this attribute to <i>SQL_FALSE</i> return <i>SQL_ERROR/SQLSTATEHYC00</i> (feature not implemented).

6.5 ステートメント属性

ODBC SQLGetStmtAttr() および SQLSetStmtAttr() 関数を使用して、ステートメント属性の値を取得および設定できます。

6.5.1 SQLGetStmtAttr ()

SQLGetStmtAttr() ファンクションは、ステートメント属性の現在の値を返します。シグネチャは次のとおりです。

```
SQLRETURN SQLGetConnectAttr
(
    SQLHDBC stmt_handle, //Input
    SQLINTEGER attribute, //Input
    SQLPOINTER value_ptr, //Output
    SQLINTEGER buffer_length, //Input
    SQLINTEGER * string_length_pointer //Output
);
```

- `stmt_handle` & mdash; 文ハンドル。
- `attribute` & mdash; 属性値。
- `value_pointer` & mdash; `attribute` 値を受け取るメモリの場所へのポインタ。
- `buffer_length` & mdash; 属性が ODBC によって定義されている場合、`buffer_length` は `value_pointer` の長さです（`value_pointer` が文字の並びまたはバイナリバッファを指している場合）。`value_pointer` が整数を指す場合、`buffer_length` は無視されます。

EDB- ODBC が属性を定義する場合、アプリケーションは `buffer_length` パラメータを設定します。`buffer_length` は次のいずれかです。

Value type	Meaning
Character string	The length of the character string
Binary buffer	The result of <i>SQL_LEN_BINARY_ATTR(length)</i>
Fixed length data type	<i>SQL_IS_INTEGER</i> or <i>SQL_IS_UIINTEGER</i>
Any other type	<i>SQL_IS_POINTER</i>

- `string_length_pointer` & mdash; 要求された値を保持するために必要なバイト数を受け取る `SQLINTEGER` へのポインタ。`value_pointer` が NULL の場合、`string_length_pointer` は返されません。

このファンクションは、`SQL_SUCCESS`、`SQL_SUCCESS_WITH_INFO`、`SQL_ERROR`、または `SQL_INVALID_HANDLE` を返します。

Attribute	Supported?	Restrictions?
SQL_ATTR_APP_PARAM_DESC	Yes	
SQL_ATTR_APP_ROW_DESC	Yes	
SQL_ATTR_ASYNC_ENABLE	No	
SQL_ATTR_CONCURRENCY	Yes	SQL_CONCUR_READ_ONLY
SQL_ATTR_CURSOR_SCROLLABLE	Yes	
SQL_ATTR_CURSOR_TYPE	Yes	SQL_CURSOR_FORWARD_ONLY
SQL_ATTR_CURSOR_SENSITIVITY	Yes	SQL_INSENSITIVE
SQL_ATTR_ENABLE_AUTO_IPD	No	
SQL_ATTR_FETCH_BOOKMARK_PTR	Yes	
SQL_ATTR_IMP_PARAM_DESC	Yes	
SQL_ATTR_IMP_ROW_DESC	Yes	
SQL_ATTR_KEYSET_SIZE	No	
SQL_ATTR_MAX_LENGTH	No	
SQL_ATTR_MAX_ROWS	No	
SQL_ATTR_METADATA_ID	Yes	
SQL_ATTR_NOSCAN	No	
SQL_ATTR_PARAM_BIND_OFFSET_PTR	Yes	ODBC V2.0
SQL_ATTR_PARAM_BIND_TYPE	Yes	
SQL_ATTR_PARAM_OPERATION_PTR	Yes	
SQL_ATTR_PARAM_STATUS_PTR	Yes	
SQL_ATTR_PARAMS_PROCESSED_PTR	Yes	
SQL_ATTR_PARAMSET_SIZE	Yes	
SQL_ATTR_QUERY_TIMEOUT	No	
SQL_ATTR_RETRIEVE_DATA	No	
SQL_ATTR_ROW_BIND_OFFSET_PTR	Yes	
SQL_ATTR_ROW_BIND_TYPE	No	
SQL_ATTR_ROW_NUMBER	Yes	
SQL_ATTR_ROW_OPERATION_PTR	Yes	
SQL_ATTR_ROW_STATUS_PTR	Yes	
SQL_ATTR_ROWS_FETCHED_PTR	Yes	
SQL_ATTR_ROW_ARRAY_SIZE	Yes	
SQL_ATTR_SIMULATE_CURSOR	No	
SQL_ATTR_USE_BOOKMARKS	Yes	
SQL_ROWSET_SIZE	Yes	

6.5.2 SQLSetStmtAttr ()

SQLSetStmtAttr() ファンクションを使用して、環境属性の値を設定できます。シグネチャは次のとおりです。

```
SQLRETURN SQLSetStmtAttr
(
    SQLHENV stmt_handle , //Input
    SQLINTEGER attribute , //Input
    SQLPOINTER value_pointer , //Input
    SQLINTEGER string_length //Input
);
```

- `stmt_handle` & mdash; 環境ハンドル。
- `attribute` & mdash; 値を設定するステートメント属性を識別します。
- `value_pointer` & mdash; 属性に割り当てられた値を保持するメモリの場所へのポインタ。
`value_pointer` は、以下へのポインタにすることができます。
- `null` で終わる文字の並び-バイナリバッファ-ドライバによって定義された値- `SQLLEN`、`SQLULEN`、
または `SQLUSMALLINT` 型の値

`value_pointer` は、オプションで次の値のいずれかを保持することもできます。- ODBC ディスクリプタ
ハンドル- `SQLINTEGER` 値- `SQLULEN` 値-署名された `INTEGER`（属性がドライバー固有の値の場合）

- `string_length` & mdash; `attribute` が ODBC によって定義され、`value_pointer` がバイナリバッ
ファ-または文字の並びを指す場合、`string_length` は `value_pointer` の長さです。`value_pointer`
が整数を指す場合、`string_length` は無視されます。EDB- ODBC が属性を定義する場合、アプ
リケーションは `string_length` パラメータを設定します。可能な `string_length` 値は次のとおり
です。

Value type	Meaning
Character string	The length of the character string or <i>SQL_NTS</i>
Binary buffer	The result of <i>SQL_LEN_BINARY_ATTR(length)</i>
Fixed length data type	<i>SQL_IS_INTEGER</i> or <i>SQL_IS_INTEGER</i>
Any other type	<i>SQL_IS_POINTER</i>

6.6 エラーハンドリング

ODBC SQLGetDiagRec() ファンクションを使用して、これらの ODBC 関数の診断情報を取得できます。

6.6.1 SQLGetDiagRec ()

SQLGetDiagRec() ファンクションは、属性値を取得または設定する ODBC 関数によって書き込まれた診断レコードからステータスとエラー情報を返します。シグネチャは次のとおりです。

```
SQLRETURN SQLGetDiagRec
(
    SQLSMALLINT handle_type, // Input
    SQLHANDLE handle, // Input
    SQLSMALLINT record_number, // Input
    SQLCHAR *SQLState_pointer, // Output
    SQLINTEGER *native_error_pointer, // Output
    SQLCHAR *error_text_pointer, // Output
    SQLSMALLINT buffer_length, // Input
    SQLSMALLINT *text_length_pointer // Output
);
```

- `handle_type` & mdash; `handle` 引数のハンドルタイプ。 `handle_type` は次のいずれかでなければなりません。
- `SQL_HANDLE_ENV` は環境ハンドルを指定します。 - `SQL_HANDLE_STMT` は、ステートメントハンドルを指定します。 - `SQL_HANDLE_DBC` は、接続ハンドルを指定します。 - `SQL_HANDLE_DESC` は、ディסקリプタハンドルを指定します。
- `handle` & mdash; 属性エラーメッセージに関連付けられたハンドル。
- `record_number` & mdash; アプリケーションが情報を探しているステータスレコード (1 以上である必要があります)。
- `SQLState_pointer` & mdash; レコードから `SQLState` エラーコードを受け取るメモリバッファへのポインタ。
- `native_error_pointer` & mdash; データソースのネイティブエラーメッセージを受信するバッファへのポインタ (`SQL_DIAG_NATIVE` フィールドに含まれる)。
- `error_text_pointer` & mdash; エラーテキスト (`SQL_DIAG_MESSAGE_TEXT` フィールドに含まれる) を受信するメモリバッファへのポインタ。
- `buffer_length` & mdash; `error_text` バッファの長さ。
- `text_length_pointer` & mdash; `error_text_pointer` フィールドのサイズ (文字数) を受け取るバッファへのポインタ。 `error_text_pointer` パラメータの文字数が使用可能な数 (`buffer_length` 内) を超える場合、 `error_text_pointer` は切り捨てられます。

SQLGetDiagRec() は、 `SQL_SUCCESS`、 `SQL_ERROR`、 `SQL_INVALID_HANDLE`、 `SQL_SUCCESS_WITH_DATA` または `SQL_NO_DATA` を返します。

6.7 サポートされている ODBC API 関数

次の表に、ODBC API 関数を示します。API が EDB- ODBC ドライバによってサポートされている場合、右の列は Yes を指定します。ODBC SQLGetFunctions() ファンクション（ファンクション ID SQL_API_ODBC3_ALL_FUNCTIONS を指定）を使用して、このリストの現在のバージョンを結果ます。

ODBC API function name	Supported by EDB-ODBC?
SQLAllocConnect()	Yes
SQLAllocEnv()	Yes
SQLAllocStmt()	Yes
SQLBindCol()	Yes
SQLCancel()	Yes
SQLColAttributes()	Yes
SQLConnect()	Yes
SQLDescribeCol()	Yes
SQLDisconnect()	Yes
SQLError()	Yes
SQLExecDirect()	Yes
SQLExecute()	Yes
SQLFetch()	Yes
SQLFreeConnect()	Yes
SQLFreeEnv()	Yes
SQLFreeStmt()	Yes
SQLGetCursorName()	Yes
SQLNumResultCols()	Yes
SQLPrepare()	Yes
SQLRowCount()	Yes
SQLSetCursorName()	Yes
SQLSetParam()	Yes
SQLTransact()	Yes
SQLColumns()	Yes
SQLDriverConnect()	Yes
SQLGetConnectOption()	Yes
SQLGetData()	Yes
SQLGetFunctions()	Yes
SQLGetInfo()	Yes
SQLGetStmtOption()	Yes
SQLGetTypeInfo()	Yes
SQLParamData()	Yes
SQLPutData()	Yes
SQLSetConnectOption()	Yes
SQLSetStmtOption()	Yes
SQLSpecialColumns()	Yes
SQLStatistics()	Yes

次のページに続く

表 11 – 前のページからの続き

ODBC API function name	Supported by EDB-ODBC?
SQLTables()	Yes
SQLBrowseConnect()	No
SQLColumnPrivileges()	No
SQLDataSources()	Yes
SQLDescribeParam()	No
SQLExtendedFetch()	Yes
SQLForeignKeys()	Yes
SQLMoreResults()	Yes
SQLNativeSQL()	Yes
SQLNumParams()	Yes
SQLParamOptions()	Yes
SQLPrimaryKeys()	Yes
SQLProcedureColumns()	Yes
SQLProcedures()	Yes
SQLSetPos()	Yes
SQLSetScrollOptions()	No
SQLTablePrivileges()	Yes
SQLDrivers()	Yes
SQLBindParameter()	Yes
SQLAllocHandle()	Yes
SQLBindParam()	Yes
SQLCloseCursor()	Yes
SQLColAttribute()	Yes
SQLCopyDesc()	Yes
SQLEndTran()	Yes
SQLFetchScroll()	Yes
SQLFreeHandle()	Yes
SQLGetConnectAttr()	Yes
SQLGetDescField()	Yes
SQLGetDescRec()	Yes
SQLGetDiagField()	Yes
SQLGetDiagRec()	Yes
SQLGetEnvAttr()	Yes
SQLGetStmtAttr()	Yes
SQLSetConnectAttr()	Yes
SQLSetDescField()	Yes
SQLSetDescRec()	No
SQLSetEnvAttr()	Yes
SQLSetStmtAttr()	Yes
SQLBulkOperations()	Yes

6.8 サポートされているデータ型

EDB- ODBC は、次の ODBC データ型をサポートしています。

ODBC data type	Corresponding EDB Postgres Advanced Server data type
SQL_BIGINT	PG_TYPE_INT8
SQL_BINARY	PG_TYPE_BYTEA
SQL_BIT	PG_TYPE_BOOL or PG_TYPE_CHAR
SQL_CHAR	PG_TYPE_BPCHAR
SQL_TYPE_DATE	PG_TYPE_DATE
SQL_DECIMAL	PG_TYPE_NUMERIC
SQL_DOUBLE	PG_TYPE_FLOAT8
SQL_FLOAT	PG_TYPE_FLOAT8
SQL_INTEGER	PG_TYPE_INT4
SQL_LONGVARBINARY	PG_TYPE_BYTEA
SQL_LONGVARCHAR	PG_TYPE_VARCHAR or PG_TYPE_TEXT
SQL_NUMERIC	PG_TYPE_NUMERIC
SQL_NUMERIC	PG_TYPE_NUMERIC
SQL_REAL	PG_TYPE_FLOAT4
SQL_SMALLINT	PG_TYPE_INT2
SQL_TYPE_TIME	PG_TYPE_TIME
SQL_TYPE_TIMESTAMP	PG_TYPE_DATETIME
SQL_TINYINT	PG_TYPE_INT2
SQL_VARBINARY	PG_TYPE_BYTEA
SQL_VARCHAR	PG_TYPE_VARCHAR

6.9 ADO ユーザーの前提条件

Command.Execute を実行前に、Command.Prepared = True を実行する必要があります。

6.10 スレッドセーフ

EDB- ODBC はスレッドセーフです。

第7章 Scram compatibility

</ div>

EDB ODBC コネクターは、EDB Postgres Advanced Server バージョン 10 以降の SCRAM-SHA-256 サポートを提供します。このサポートは、EDB ODBC 10.01.0000.01 リリース以降で利用可能です。

第8章 SLES 15 on IBM Power (ppc64le)

インストールを完了するには2つの手順があります。

- リポジトリのセットアップ
- パッケージのインストール

各ステップで、スーパーユーザとしてログインする必要があります。

```
# To log in as a superuser:  
sudo su -
```

リポジトリを設定する前に、EDBに登録する必要があります。EDB リポジトリの資格情報を受信するには、[Repository Access Request](#) にアクセスしてください。

8.1 リポジトリのセットアップ

リポジトリの設定は1回限りのタスクです。リポジトリをすでに設定している場合、これらの手順を実行する必要はありません。

```
# Install the repository configuration and enter your EDB repository  
# credentials when prompted  
zypper addrepo https://zypp.enterprisedb.com/suse/edb-sles.repo  
  
# Install SUSEConnect to register the host with SUSE, allowing access to  
# SUSE repositories  
zypper install SUSEConnect  
  
# Register the host with SUSE, allowing access to SUSE repositories  
# Replace REGISTRATION_CODE and EMAIL with your SUSE registration  
# information  
SUSEConnect -r REGISTRATION_CODE -e EMAIL  
  
# Activate the required SUSE module  
SUSEConnect -p PackageHub/15.3/ppc64le  
  
# Refresh the metadata  
zypper refresh
```

8.2 パッケージのインストール

```
zypper -n install edb-odbc
```

第9章 Upgrading a Linux installation

既存の EDB ODBC コネクタ RPM インストールがある場合、yum または dnf を使用して、リポジトリ構成ファイルをアップグレードし、より新しい製品バージョンに更新できます。edb.repo ファイルを更新するには、スーパーユーザ権限を引き受けて次のように入力します。

- RHEL または CentOS 7 の場合：

```
yum upgrade edb-repo
```

- RHEL または Rocky Linux または AlmaLinux 8 の場合：

```
dnf upgrade edb-repo
```

yum または dnf は、edb.repo ファイルを更新して、edb.repo ファイルで指定された資格情報で接続するように構成された現在の EDB リポジトリにアクセスできるようにします。次に、yum または dnf を使用して、インストールされているパッケージソフトをアップグレードできます。

- RHEL または CentOS 7 の場合：

```
yum upgrade edb-odbc
```

```
yum upgrade edb-odbc-devel
```

- RHEL または Rocky Linux または AlmaLinux 8 の場合：

```
dnf upgrade edb-odbc
```

```
dnf upgrade edb-odbc-devel
```