TDS Foreign Data Wrapper

Version 2

Built at 2026-05-22T16:07:30

1 TDS Foreign Data Wrapper

TDS foreign data wrapper (`tds_fdw`) lets you connect Postgres to databases that use the Tabular Data Stream (TDS) protocol, such as Sybase databases and Microsoft SQL Server. It uses the FreeTDS library to implement the DB-Library interface.

Architecture overview

TDS foreign data wrapper provides an interface between a TDS-compatible database server (such as Microsoft SQL Server or Sybase) and a Postgres database. It transforms Postgres statements (`SELECT`) into queries that are understood by the remote TDS database, using the FreeTDS library to implement the DB-Library interface.

The wrapper supports WHERE clause and column push downs, which means filters and column selections are evaluated on the remote server rather than locally.

Key features

These are the key features of TDS foreign data wrapper.

WHERE clause pushdown

TDS foreign data wrapper supports pushdown of the `WHERE` clause. Filter conditions are evaluated on the remote TDS server rather than fetching all rows locally.

For more information, see Foreign table options.

Column pushdown

TDS foreign data wrapper supports column pushdown. Only the columns required by the query are fetched from the remote server, reducing data transfer.

For more information, see Foreign table options.

Import foreign schema

TDS foreign data wrapper supports `IMPORT FOREIGN SCHEMA` , which lets you import an entire remote schema rather than defining each foreign table manually.

For more information, see Import a foreign schema.

FreeTDS connection pooling

TDS foreign data wrapper establishes a connection to a foreign server during the first query that uses a foreign table associated with the foreign server. This connection is kept and reused for subsequent queries in the same session.

Automated cleanup

TDS foreign data wrapper allows the cleanup of foreign tables and server objects using the `DROP EXTENSION` command:

```
DROP EXTENSION tds_fdw CASCADE;
```

Flexible remote execution

TDS foreign data wrapper supports both table-based and query-based foreign tables. You can define a foreign table that maps to a remote table or view, or one that executes a custom SQL query on the remote server.

For more information, see Create a foreign table.

2 Release notes

The TDS foreign data wrapper documentation describes the latest version of TDS Foreign Data Wrapper 2, including minor releases and patches. The release notes provide information on what was new in each release. For new functionality introduced in a minor or patch release, the content also indicates the release that introduced the feature.

Version	Release date
2.0.5	19 May 2026

2.1 TDS Foreign Data Wrapper 2.0.5 release notes

Released: 19 May 2026

Enhancements, bug fixes, and other changes in TDS Foreign Data Wrapper 2.0.5 include:

Type	Description
Enhancement	Added support for PostgreSQL, EDB Postgres Advanced Server, and EDB Postgres Extended Server 18.
Enhancement	Updated `tds_fdw.c` to improve data type mapping (handling `money` to `decimal`).
Enhancement	Updated the internal testing matrix for better platform stability.

3 Supported platforms

TDS foreign data wrapper is supported on the same platforms as EDB Postgres Advanced Server, with the exception of SLES 15. To determine the platform support for TDS foreign data wrapper, see the Platform Compatibility page on the EDB website or Installing TDS foreign data wrapper.

Supported database versions

This table lists the latest TDS foreign data wrapper versions and their supported corresponding EDB Postgres Advanced Server (EPAS), EDB Postgres Extended Server (PGE), and PostgreSQL (PG) versions.

TDS foreign data wrapper	EPAS, PGE, PG 18	EPAS, PGE, PG 17	EPAS, PGE, PG 16	EPAS, PGE, PG 15	EPAS, PGE, PG 14
2.0.5	Y	Y	Y	Y	Y

4 Limitations

The following limitations apply to TDS foreign data wrapper:

- TDS foreign data wrapper doesn't currently support `JOIN` pushdown.
- TDS foreign data wrapper doesn't currently support write operations (`INSERT` , `UPDATE` , `DELETE`).
- The `CREATE FOREIGN TABLE` option `row_estimate_method` supports value `showplan_all` only with Microsoft SQL Server. It isn't available for Sybase databases.

5 Installing TDS Foreign Data Wrapper on Linux

Select a link to access the applicable installation instructions:

Linux x86-64 (amd64)

Red Hat Enterprise Linux (RHEL) and derivatives

- RHEL 9, RHEL 8
- Oracle Linux (OL) 9, Oracle Linux (OL) 8
- Rocky Linux 9, Rocky Linux 8
- AlmaLinux 9, AlmaLinux 8

Debian and derivatives

- Ubuntu 24.04, Ubuntu 22.04
- Debian 12, Debian 11

Linux IBM Power (ppc64le)

Red Hat Enterprise Linux (RHEL) and derivatives

- RHEL 9, RHEL 8

Linux AArch64 (ARM64)

Red Hat Enterprise Linux (RHEL) and derivatives

- RHEL 9
- Oracle Linux (OL) 9

Debian and derivatives

- Debian 12

5.1 Installing TDS Foreign Data Wrapper on Linux x86-64 (amd64)

Select a link to access the applicable installation instructions.

Red Hat Enterprise Linux (RHEL) and derivatives

- RHEL 9 or OL 9
- RHEL 8 or OL 8
- AlmaLinux 9 or Rocky Linux 9
- AlmaLinux 8 or Rocky Linux 8

Debian and derivatives

- Ubuntu 24.04
- Ubuntu 22.04
- Debian 12
- Debian 11

5.1.1 Installing TDS Foreign Data Wrapper on RHEL 9 or OL 9 x86_64

Prerequisites

Before you begin the installation process:

- Install Postgres on the same host. See:
  - Installing EDB Postgres Advanced Server
  - Installing PostgreSQL
- Set up the EDB repository.

  Setting up the repository is a one-time task. If you already set up your repository, you don't need to perform this step.

  To determine if your repository exists, enter:

  ```
  dnf repolist | grep enterprisedb
  ```

  If no output is generated, the repository is installed.

  To set up the EDB repository:

  1. Go to EDB repositories.
  2. Select the button that provides access to the EDB repository.
  3. Select the platform and software that you want to download.
  4. Follow the instructions for setting up the EDB repository.
- Install the EPEL repository:

  ```
  sudo dnf -y install https://dl.fedoraproject.org/pub/epel/epel-release-latest-9.noarch.rpm
  ```
- Install the FreeTDS library:

  ```
  sudo dnf -y install freetds-devel
  ```

Install the package

```
# For EDB Postgres Advanced Server:
sudo dnf -y install edb-as<X>-tds-fdw

# For EDB Postgres Extended Server:
sudo dnf -y install edb-postgresextended<X>-tds-fdw

# For PostgreSQL:
sudo dnf -y install edb-pg<X>-tds-fdw
```

Where `<X>` is the version of your Postgres.

5.1.2 Installing TDS Foreign Data Wrapper on RHEL 8 or OL 8 x86_64

Prerequisites

Before you begin the installation process:

- Install Postgres on the same host. See:
  - Installing EDB Postgres Advanced Server
  - Installing PostgreSQL
- Set up the EDB repository.

  Setting up the repository is a one-time task. If you already set up your repository, you don't need to perform this step.

  To determine if your repository exists, enter:

  ```
  dnf repolist | grep enterprisedb
  ```

  If no output is generated, the repository is installed.

  To set up the EDB repository:

  1. Go to EDB repositories.
  2. Select the button that provides access to the EDB repository.
  3. Select the platform and software that you want to download.
  4. Follow the instructions for setting up the EDB repository.
- Install the EPEL repository:

  ```
  sudo dnf -y install https://dl.fedoraproject.org/pub/epel/epel-release-latest-8.noarch.rpm
  ```
- Install the FreeTDS library:

  ```
  sudo dnf -y install freetds-devel
  ```

Install the package

```
# For EDB Postgres Advanced Server:
sudo dnf -y install edb-as<X>-tds-fdw

# For EDB Postgres Extended Server:
sudo dnf -y install edb-postgresextended<X>-tds-fdw

# For PostgreSQL:
sudo dnf -y install edb-pg<X>-tds-fdw
```

Where `<X>` is the version of your Postgres.

5.1.3 Installing TDS Foreign Data Wrapper on AlmaLinux 9 or Rocky Linux 9 x86_64

Prerequisites

Before you begin the installation process:

- Install Postgres on the same host. See:
  - Installing EDB Postgres Advanced Server
  - Installing PostgreSQL
- Set up the EDB repository.

  Setting up the repository is a one-time task. If you already set up your repository, you don't need to perform this step.

  To determine if your repository exists, enter:

  ```
  dnf repolist | grep enterprisedb
  ```

  If no output is generated, the repository is installed.

  To set up the EDB repository:

  1. Go to EDB repositories.
  2. Select the button that provides access to the EDB repository.
  3. Select the platform and software that you want to download.
  4. Follow the instructions for setting up the EDB repository.
- Install the EPEL repository:

  ```
  sudo dnf -y install epel-release
  ```
- Enable additional repositories to resolve dependencies:

  ```
  sudo dnf config-manager --set-enabled crb
  ```
- Install the FreeTDS library:

  ```
  sudo dnf -y install freetds-devel
  ```

Install the package

```
# For EDB Postgres Advanced Server:
sudo dnf -y install edb-as<X>-tds-fdw

# For EDB Postgres Extended Server:
sudo dnf -y install edb-postgresextended<X>-tds-fdw

# For PostgreSQL:
sudo dnf -y install edb-pg<X>-tds-fdw
```

Where `<X>` is the version of your Postgres.

5.1.4 Installing TDS Foreign Data Wrapper on AlmaLinux 8 or Rocky Linux 8 x86_64

Prerequisites

Before you begin the installation process:

- Install Postgres on the same host. See:
  - Installing EDB Postgres Advanced Server
  - Installing PostgreSQL
- Set up the EDB repository.

  Setting up the repository is a one-time task. If you already set up your repository, you don't need to perform this step.

  To determine if your repository exists, enter:

  ```
  dnf repolist | grep enterprisedb
  ```

  If no output is generated, the repository is installed.

  To set up the EDB repository:

  1. Go to EDB repositories.
  2. Select the button that provides access to the EDB repository.
  3. Select the platform and software that you want to download.
  4. Follow the instructions for setting up the EDB repository.
- Install the EPEL repository:

  ```
  sudo dnf -y install epel-release
  ```
- Enable additional repositories to resolve dependencies:

  ```
  sudo dnf config-manager --set-enabled powertools
  ```
- Install the FreeTDS library:

  ```
  sudo dnf -y install freetds-devel
  ```

Install the package

```
# For EDB Postgres Advanced Server:
sudo dnf -y install edb-as<X>-tds-fdw

# For EDB Postgres Extended Server:
sudo dnf -y install edb-postgresextended<X>-tds-fdw

# For PostgreSQL:
sudo dnf -y install edb-pg<X>-tds-fdw
```

Where `<X>` is the version of your Postgres.

5.1.5 Installing TDS Foreign Data Wrapper on Ubuntu 24.04 x86_64

Prerequisites

Before you begin the installation process:

- Install Postgres on the same host. See:
  - Installing EDB Postgres Advanced Server
  - Installing PostgreSQL
- Set up the EDB repository.

  Setting up the repository is a one-time task. If you already set up your repository, you don't need to perform this step.

  To determine if your repository exists, enter:

  ```
  apt-cache search enterprisedb
  ```

  If no output is generated, the repository is installed.

  To set up the EDB repository:

  1. Go to EDB repositories.
  2. Select the button that provides access to the EDB repository.
  3. Select the platform and software that you want to download.
  4. Follow the instructions for setting up the EDB repository.
- Install the FreeTDS library:

  ```
  sudo apt-get -y install freetds-dev
  ```

Install the package

```
# For EDB Postgres Advanced Server:
sudo apt-get -y install edb-as<X>-tds-fdw

# For EDB Postgres Extended Server:
sudo apt-get -y install edb-postgresextended<X>-tds-fdw

# For PostgreSQL:
sudo apt-get -y install edb-pg<X>-tds-fdw
```

Where `<X>` is the version of your Postgres.

5.1.6 Installing TDS Foreign Data Wrapper on Ubuntu 22.04 x86_64

Prerequisites

Before you begin the installation process:

- Install Postgres on the same host. See:
  - Installing EDB Postgres Advanced Server
  - Installing PostgreSQL
- Set up the EDB repository.

  Setting up the repository is a one-time task. If you already set up your repository, you don't need to perform this step.

  To determine if your repository exists, enter:

  ```
  apt-cache search enterprisedb
  ```

  If no output is generated, the repository is installed.

  To set up the EDB repository:

  1. Go to EDB repositories.
  2. Select the button that provides access to the EDB repository.
  3. Select the platform and software that you want to download.
  4. Follow the instructions for setting up the EDB repository.
- Install the FreeTDS library:

  ```
  sudo apt-get -y install freetds-dev
  ```

Install the package

```
# For EDB Postgres Advanced Server:
sudo apt-get -y install edb-as<X>-tds-fdw

# For EDB Postgres Extended Server:
sudo apt-get -y install edb-postgresextended<X>-tds-fdw

# For PostgreSQL:
sudo apt-get -y install edb-pg<X>-tds-fdw
```

Where `<X>` is the version of your Postgres.

5.1.7 Installing TDS Foreign Data Wrapper on Debian 12 x86_64

Prerequisites

Before you begin the installation process:

- Install Postgres on the same host. See:
  - Installing EDB Postgres Advanced Server
  - Installing PostgreSQL
- Set up the EDB repository.

  Setting up the repository is a one-time task. If you already set up your repository, you don't need to perform this step.

  To determine if your repository exists, enter:

  ```
  apt-cache search enterprisedb
  ```

  If no output is generated, the repository is installed.

  To set up the EDB repository:

  1. Go to EDB repositories.
  2. Select the button that provides access to the EDB repository.
  3. Select the platform and software that you want to download.
  4. Follow the instructions for setting up the EDB repository.
- Install the FreeTDS library:

  ```
  sudo apt-get -y install freetds-dev
  ```

Install the package

```
# For EDB Postgres Advanced Server:
sudo apt-get -y install edb-as<X>-tds-fdw

# For EDB Postgres Extended Server:
sudo apt-get -y install edb-postgresextended<X>-tds-fdw

# For PostgreSQL:
sudo apt-get -y install edb-pg<X>-tds-fdw
```

Where `<X>` is the version of your Postgres.

5.1.8 Installing TDS Foreign Data Wrapper on Debian 11 x86_64

Prerequisites

Before you begin the installation process:

- Install Postgres on the same host. See:
  - Installing EDB Postgres Advanced Server
  - Installing PostgreSQL
- Set up the EDB repository.

  Setting up the repository is a one-time task. If you already set up your repository, you don't need to perform this step.

  To determine if your repository exists, enter:

  ```
  apt-cache search enterprisedb
  ```

  If no output is generated, the repository is installed.

  To set up the EDB repository:

  1. Go to EDB repositories.
  2. Select the button that provides access to the EDB repository.
  3. Select the platform and software that you want to download.
  4. Follow the instructions for setting up the EDB repository.
- Install the FreeTDS library:

  ```
  sudo apt-get -y install freetds-dev
  ```

Install the package

```
# For EDB Postgres Advanced Server:
sudo apt-get -y install edb-as<X>-tds-fdw

# For EDB Postgres Extended Server:
sudo apt-get -y install edb-postgresextended<X>-tds-fdw

# For PostgreSQL:
sudo apt-get -y install edb-pg<X>-tds-fdw
```

Where `<X>` is the version of your Postgres.

5.2 Installing TDS Foreign Data Wrapper on Linux AArch64 (ARM64)

Select a link to access the applicable installation instructions.

Red Hat Enterprise Linux (RHEL) and derivatives

- RHEL 9 or OL 9

Debian and derivatives

- Debian 12

5.2.1 Installing TDS Foreign Data Wrapper on RHEL 9 or OL 9 ARM64

Prerequisites

Before you begin the installation process:

- Install Postgres on the same host. See:
  - Installing EDB Postgres Advanced Server
  - Installing PostgreSQL
- Set up the EDB repository.

  Setting up the repository is a one-time task. If you already set up your repository, you don't need to perform this step.

  To determine if your repository exists, enter:

  ```
  dnf repolist | grep enterprisedb
  ```

  If no output is generated, the repository is installed.

  To set up the EDB repository:

  1. Go to EDB repositories.
  2. Select the button that provides access to the EDB repository.
  3. Select the platform and software that you want to download.
  4. Follow the instructions for setting up the EDB repository.
- Install the EPEL repository:

  ```
  sudo dnf -y install https://dl.fedoraproject.org/pub/epel/epel-release-latest-9.noarch.rpm
  ```
- Install the FreeTDS library:

  ```
  sudo dnf -y install freetds-devel
  ```

Install the package

```
# For EDB Postgres Advanced Server:
sudo dnf -y install edb-as<X>-tds-fdw

# For EDB Postgres Extended Server:
sudo dnf -y install edb-postgresextended<X>-tds-fdw

# For PostgreSQL:
sudo dnf -y install edb-pg<X>-tds-fdw
```

Where `<X>` is the version of your Postgres.

5.2.2 Installing TDS Foreign Data Wrapper on Debian 12 ARM64

Prerequisites

Before you begin the installation process:

- Install Postgres on the same host. See:
  - Installing EDB Postgres Advanced Server
  - Installing PostgreSQL
- Set up the EDB repository.

  Setting up the repository is a one-time task. If you already set up your repository, you don't need to perform this step.

  To determine if your repository exists, enter:

  ```
  apt-cache search enterprisedb
  ```

  If no output is generated, the repository is installed.

  To set up the EDB repository:

  1. Go to EDB repositories.
  2. Select the button that provides access to the EDB repository.
  3. Select the platform and software that you want to download.
  4. Follow the instructions for setting up the EDB repository.
- Install the FreeTDS library:

  ```
  sudo apt-get -y install freetds-dev
  ```

Install the package

```
# For EDB Postgres Advanced Server:
sudo apt-get -y install edb-as<X>-tds-fdw

# For EDB Postgres Extended Server:
sudo apt-get -y install edb-postgresextended<X>-tds-fdw

# For PostgreSQL:
sudo apt-get -y install edb-pg<X>-tds-fdw
```

Where `<X>` is the version of your Postgres.

5.3 Installing TDS Foreign Data Wrapper on Linux IBM Power (ppc64le)

Select a link to access the applicable installation instructions.

Red Hat Enterprise Linux (RHEL) and derivatives

- RHEL 9
- RHEL 8

5.3.1 Installing TDS Foreign Data Wrapper on RHEL 9 ppc64le

Prerequisites

Before you begin the installation process:

- Install Postgres on the same host. See:
  - Installing EDB Postgres Advanced Server
  - Installing PostgreSQL
- Set up the EDB repository.

  Setting up the repository is a one-time task. If you already set up your repository, you don't need to perform this step.

  To determine if your repository exists, enter:

  ```
  dnf repolist | grep enterprisedb
  ```

  If no output is generated, the repository is installed.

  To set up the EDB repository:

  1. Go to EDB repositories.
  2. Select the button that provides access to the EDB repository.
  3. Select the platform and software that you want to download.
  4. Follow the instructions for setting up the EDB repository.
- Install the EPEL repository:

  ```
  sudo dnf -y install https://dl.fedoraproject.org/pub/epel/epel-release-latest-9.noarch.rpm
  ```
- Install the FreeTDS library:

  ```
  sudo dnf -y install freetds-devel
  ```

Install the package

```
# For EDB Postgres Advanced Server:
sudo dnf -y install edb-as<X>-tds-fdw

# For EDB Postgres Extended Server:
sudo dnf -y install edb-postgresextended<X>-tds-fdw

# For PostgreSQL:
sudo dnf -y install edb-pg<X>-tds-fdw
```

Where `<X>` is the version of your Postgres.

5.3.2 Installing TDS Foreign Data Wrapper on RHEL 8 ppc64le

Prerequisites

Before you begin the installation process:

- Install Postgres on the same host. See:
  - Installing EDB Postgres Advanced Server
  - Installing PostgreSQL
- Set up the EDB repository.

  Setting up the repository is a one-time task. If you already set up your repository, you don't need to perform this step.

  To determine if your repository exists, enter:

  ```
  dnf repolist | grep enterprisedb
  ```

  If no output is generated, the repository is installed.

  To set up the EDB repository:

  1. Go to EDB repositories.
  2. Select the button that provides access to the EDB repository.
  3. Select the platform and software that you want to download.
  4. Follow the instructions for setting up the EDB repository.
- Install the EPEL repository:

  ```
  sudo dnf -y install https://dl.fedoraproject.org/pub/epel/epel-release-latest-8.noarch.rpm
  ```
- Install the FreeTDS library:

  ```
  sudo dnf -y install freetds-devel
  ```

Install the package

```
# For EDB Postgres Advanced Server:
sudo dnf -y install edb-as<X>-tds-fdw

# For EDB Postgres Extended Server:
sudo dnf -y install edb-postgresextended<X>-tds-fdw

# For PostgreSQL:
sudo dnf -y install edb-pg<X>-tds-fdw
```

Where `<X>` is the version of your Postgres.

6 Performing initial configuration

Before using TDS foreign data wrapper:

1. Use the CREATE EXTENSION command to create the TDS foreign data wrapper extension on the Postgres host.
2. Optionally, Configure FreeTDS for named server entries and character encoding.
3. Use the CREATE FOREIGN SERVER command to create the foreign server that defines a connection to the TDS database.
4. Use the CREATE USER MAPPING command to define a mapping that associates a Postgres role with the server.
5. Use the CREATE FOREIGN TABLE command to define a table in the Postgres database that corresponds to a table that resides on the remote TDS server.

Creating the extension

Use the `CREATE EXTENSION` command to create the `tds_fdw` extension. Connect to the database from which you want to query the TDS server, and invoke the command:

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

Parameters

`IF NOT EXISTS`

Include the `IF NOT EXISTS` clause to instruct the server to issue a notice instead of returning an error if an extension with the same name already exists.

`WITH`

A noise word included for readability. It has no effect on the command.

`schema_name`

Optionally specify the name of the schema in which to install the extension's objects.

The `CREATE EXTENSION` command installs the TDS foreign data wrapper and its associated objects in the database. You only need to run this command once per database. It creates a default `freetds.conf` during installation — you don't need to create it from scratch. The file typically lives at `/etc/freetds/freetds.conf` (or `/usr/local/etc/freetds.conf` if built from source), and it contains some default entries and comments.

For more information about using the `CREATE EXTENSION` command, see the PostgreSQL documentation.

Configuring FreeTDS

`tds_fdw` uses FreeTDS to connect to TDS databases. Configuring FreeTDS is optional — you only need it if you want to use named server entries or set the client character encoding. You can configure connection parameters either directly in the foreign server definition or in the FreeTDS configuration file `freetds.conf` (created during FreeTDS installation).

To configure a named server entry in `freetds.conf`:

```
[mssql_server]
    host = 192.168.1.100
    port = 1433
    tds version =
7.4
    client charset = UTF-
8
```

You can then refer to this entry by its section name (`mssql_server`) as the `servername` option in your foreign server definition.

Character sets and encoding

If you encounter errors like the following when working with Unicode data on Microsoft SQL Server:

```
NOTICE: DB-Library notice: Msg #: 4004, Msg state: 1, Msg: Unicode data in a
Unicode-only collation or ntext data cannot be sent to clients using DB-Library
```

You may need to set `tds version` to `7.0` or higher in `freetds.conf`. See Choosing a TDS protocol version.

To set the client character set, set `client charset` in `freetds.conf`. See Localization and TDS 7.0.

Encrypted connections to Microsoft SQL Server

Encrypted connections are handled by FreeTDS. To enable encryption, configure the `encryption` setting in `freetds.conf`. See the FreeTDS freetds.conf reference for details.

For more information on FreeTDS configuration, see the FreeTDS documentation.

Creating the foreign server

Use the `CREATE SERVER` command to define a connection to a foreign server. The syntax is:

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

The role that defines the foreign server is the owner of the foreign server. To create a foreign server, you must have `USAGE` privilege on the foreign-data wrapper specified in the `CREATE SERVER` command. If you want to change the owner of an existing foreign server, use the `ALTER SERVER` command to reassign ownership.

Parameters

`server_name`

Use `server_name` to specify a name for the foreign server. The server name must be unique in the database.

`FOREIGN DATA WRAPPER`

Include the `FOREIGN DATA WRAPPER` clause to specify that the server uses the `tds_fdw` foreign data wrapper when connecting to the TDS database.

`OPTIONS`

Use the `OPTIONS` clause to specify connection information for the foreign server object. You can include these options:

Option	Required	Default	Description
`servername`	Yes	`127.0.0.1`	The hostname, IP address, or DSN of the remote server. This can be a DSN as specified in `freetds.conf` — see FreeTDS name lookup. You can supply a comma-separated list of server names for automatic failover to a secondary server.
`port`	No	—	The port of the remote server. Can also be specified in `freetds.conf` if servername is a DSN.
`database`	No	—	The database to connect to on the remote server.
`dbuse`	No	`0`	If `0`, connects directly to `database`. If non-zero, connects to the server's default database and then selects `database` using `dbuse()` function. Set to `0` for Azure SQL Database.
`language`	No	—	The language for messages and date format locale. Defaults to `us_english` in most FreeTDS installations. It can be changed in `freetds.conf`. In MS SQL Server, see SET LANGUAGE to change the language. In Sybase ASE, see SET LANGUAGE to change the language.
`character_set`	No	—	The client character set for the connection. Has no effect for TDS protocol version 7.0 and higher, which always use UCS-2. See Localization and TDS 7.0.
`tds_version`	No	—	The TDS protocol version to use. See Choosing a TDS protocol version and History of TDS versions.
`msg_handler`	No	`blackhole`	How to handle TDS messages. Use `notice` to surface them as PostgreSQL notices, or `blackhole` to discard them.

Option	Required	Default	Description
`fdw_startup_cost`	No	—	An estimated cost representing the overhead of using this FDW, used in query planning.
`fdw_tuple_cost`	No	—	An estimated cost representing the overhead of fetching rows from this server, used in query planning.
`sqlserver_ansi_mode`	No	`false`	SQL Server only. When set to `true`, enables `CONCAT_NULLS_YIELDS_NULL`, `ANSI_NULLS`, `ANSI_WARNINGS`, `QUOTED_IDENTIFIER`, `ANSI_PADDING`, and `ANSI_NULL_DFLT_ON` after connecting. These parameters are comparable to SQL Server option `ANSI_DEFAULTS`. In contrast, `sqlserver_ansi_mode` doesn't activate `CURSOR_CLOSE_ON_COMMIT` and `IMPLICIT_TRANSACTIONS` options.
`use_remote_estimate`	No	—	Whether to estimate table size by querying the remote server. Can also be set on individual foreign tables.
`row_estimate_method`	No	`execute`	Method used for remote row estimation. Can also be set on individual foreign tables.

Example

The following command creates a foreign server named `mssql_svr` that uses the `tds_fdw` foreign data wrapper to connect to a Microsoft SQL Server:

```
CREATE SERVER
mssql_svr
    FOREIGN DATA WRAPPER tds_fdw
    OPTIONS (servername '127.0.0.1', port '1433', database 'mydb', tds_version
'7.4');
```

For more information about using the `CREATE SERVER` command, see the PostgreSQL documentation.

Creating a user mapping

Use the `CREATE USER MAPPING` command to define a mapping that associates a Postgres role with a foreign server:

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

The role must be the owner of the foreign server to create a user mapping for that server.

Parameters

`role_name`

Use `role_name` to specify the role to associate with the foreign server.

`server_name`

Use `server_name` to specify the name of the server that defines a connection to the TDS database.

`OPTIONS`

Use the `OPTIONS` clause to specify connection information for the foreign server.

Option	Required	Description
`username`	Yes	The username on the remote server. For Azure SQL Database, use the format `username@servername`. For other databases, use `username`.
`password`	Yes	The password for the remote account.

Example

The following command creates a user mapping for a role named `postgres`. The mapping is associated with a server named `mssql_svr`.

```
CREATE USER MAPPING FOR
postgres
    SERVER
mssql_svr
    OPTIONS (username 'sa', password 'yourpassword');
```

For more information about using the `CREATE USER MAPPING` command, see the PostgreSQL documentation.

Creating a foreign table

A foreign table is a pointer to a table that resides on the TDS host. Use the `CREATE FOREIGN TABLE` command to define a table in the Postgres database that corresponds to a table on the remote TDS server. The syntax is:

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

Parameters

`table_name`

Specify the name of the foreign table. Include a schema name to specify the schema in which the foreign table resides.

`IF NOT EXISTS`

Include the `IF NOT EXISTS` clause to instruct the server to not return an error if a table with the same name already exists.

`column_name`

Specify the name of every column in the new table. Each column must correspond to a column on the remote TDS server.

`data_type`

Specify the data type of the column.

`server_name`

Specify the name of the foreign server that defines the connection to the TDS database.

`OPTIONS`

Use the `OPTIONS` clause to specify the following options:

Option	Required	Default	Description
`table_name`	Yes (or `query`)	—	The table on the remote server to query. The schema can be included in the name or set separately with `schema_name`. Alias: `table`.
`query`	Yes (or `table_name`)	—	A SQL query string to execute on the remote server instead of reading a table directly.
`schema_name`	No	—	The schema the remote table is in. Can also be included in `table_name`.
`match_column_names`	No	Enabled	When enabled, local columns are matched to remote columns by name rather than by position.
`use_remote_estimate`	No	—	Whether to estimate the table size by querying the remote server.

Option	Required	Default	Description
`local_tuple_estimate`	No	—	A locally set row count estimate, used when `use_remote_estimate` is disabled.
`row_estimate_method`	No	`execute`	How to estimate row counts: `execute` runs the query and counts actual rows; `showplan_all` uses MS SQL Server's `SET SHOWPLAN_ALL` to get an estimated count without executing.

Foreign table column options

Option	Required	Description
`column_name`	No	The name of the corresponding column on the remote server. If not set, the local column name is used. Ignored when `match_column_names` is `0`.

Example

```
CREATE FOREIGN TABLE mssql_table
(
    id integer,
    data varchar)
    SERVER
mssql_svr
    OPTIONS (schema_name 'dbo', table_name 'mytable', row_estimate_method
'showplan_all');
```

For more information about using the `CREATE FOREIGN TABLE` command, see the PostgreSQL documentation.

Dropping the extension

Use the `DROP EXTENSION` command to remove the TDS foreign data wrapper extension. Connect to the Postgres database from which you're dropping TDS foreign data wrapper, and run the command:

```
DROP EXTENSION [ IF EXISTS ] name [, ...] [ CASCADE | RESTRICT
];
```

Example

```
DROP EXTENSION
tds_fdw;
```

For more information about using the `DROP EXTENSION` command, see the PostgreSQL documentation.

Dropping the server

Use the `DROP SERVER` command to remove a connection to the foreign server. The syntax is:

```
DROP SERVER [ IF EXISTS ] name [, ...] [ CASCADE | RESTRICT
];
```

Example

```
DROP SERVER
mssql_svr;
```

For more information about using the `DROP SERVER` command, see the PostgreSQL documentation.

Dropping the user mapping

Use the `DROP USER MAPPING` command to remove a mapping that associates a Postgres role with a foreign server. The user must be the owner of the foreign server to remove a user mapping for that server.

```
DROP USER MAPPING [ IF EXISTS ] FOR { user_name | USER | CURRENT_USER | PUBLIC } SERVER
server_name;
```

Example

```
DROP USER MAPPING FOR postgres SERVER
mssql_svr;
```

For more information about using the `DROP USER MAPPING` command, see the PostgreSQL documentation.

Dropping the foreign table

Use the `DROP FOREIGN TABLE` command to remove a foreign table. Only the owner of the foreign table can drop it.

```
DROP FOREIGN TABLE [ IF EXISTS ] name [, ...] [ CASCADE | RESTRICT
];
```

Example

```
DROP FOREIGN TABLE mssql_table;
```

For more information about using the `DROP FOREIGN TABLE` command, see the PostgreSQL documentation.

7 Upgrading

If you have an existing installation of TDS foreign data wrapper that you installed using the EDB repository, you can update your repository configuration file and then upgrade TDS foreign data wrapper to a more recent product version.

To perform the process, open a terminal window and enter the commands that apply to the operating system and package manager used for the installation.

To update your repository configuration file:

```
sudo <package-manager> upgrade edb-repo
```

Where `<package-manager>` is the package manager used with your operating system:

Package manager	Operating system
dnf	RHEL 8/9 and derivatives
apt-get	Debian and Ubuntu

To upgrade to the latest product version, enter one of the following commands:

Operating system	Upgrade command
RHEL 8/9 and derivatives	`sudo dnf upgrade edb-as<xx>-tds-fdw`
Debian and Ubuntu	`sudo apt-get --only-upgrade install edb-as<xx>-tds-fdw`

Where:

- `<package-manager>` is the package manager used with your operating system.
- `<xx>` is the EDB Postgres Advanced Server version number.

After upgrading the package, update the extension in each database where it's installed. Connect to your Postgres server and enter:

```
ALTER EXTENSION tds_fdw UPDATE;
```

To update to a specific version, enter:

```
ALTER EXTENSION tds_fdw UPDATE TO '<version>';
```

Where `<version>` is the version to update to, for example, `2.0.5`.

8 Uninstalling

You can use the `remove` command to uninstall TDS foreign data wrapper packages. To uninstall, open a terminal window, assume superuser privileges, and enter the command that applies to the operating system and package manager used for the installation. `xx` is the EDB Postgres Advanced Server version number.

- On RHEL or Rocky Linux or AlmaLinux 8/9:

  ```
  dnf remove edb-as<xx>-tds-fdw
  ```

- On Debian or Ubuntu:

  ```
  apt-get remove edb-as<xx>-tds-fdw
  ```

9 Using TDS foreign data wrapper

This example shows how to use `tds_fdw` to query a table on a Microsoft SQL Server from a Postgres database.

Prerequisites

- `tds_fdw` is installed and the extension is created on your Postgres database. See Create extension.
- A foreign server and user mapping are defined. See Create server and Create user mapping.

Creating a foreign table

Using a table name

To map a remote table to a foreign table on the Postgres server:

```
CREATE FOREIGN TABLE mssql_table
(
    id integer,
    data varchar)
    SERVER
mssql_svr
    OPTIONS (schema_name 'dbo', table_name 'mytable', row_estimate_method
'showplan_all');
```

Using a query

To map a custom SQL query to a foreign table:

```
CREATE FOREIGN TABLE mssql_table
(
    id integer,
    data varchar)
    SERVER
mssql_svr
    OPTIONS (query 'SELECT * FROM dbo.mytable', row_estimate_method
'showplan_all');
```

Mapping a remote column name

If a local column name differs from the remote column name, use the `column_name` option:

```
CREATE FOREIGN TABLE mssql_table
(
    id integer,
    local_col varchar OPTIONS (column_name 'remote_col'))
    SERVER
mssql_svr
    OPTIONS (schema_name 'dbo', table_name 'mytable');
```

Querying a foreign table

Once a foreign table is set up, you can query it like any local table:

```
SELECT * FROM mssql_table WHERE id =
42;
```

`tds_fdw` supports `WHERE` clause pushdown and column pushdown, which means filters and column selections are evaluated on the remote server rather than locally.

`tds_fdw` doesn't currently support `JOIN` pushdown or write operations (`INSERT` , `UPDATE` , `DELETE`).

Importing a foreign schema

You can import an entire remote schema rather than defining each foreign table manually:

```
IMPORT FOREIGN SCHEMA
dbo
    EXCEPT (mssql_table)
    FROM SERVER
mssql_svr
    INTO public
    OPTIONS (import_default 'true');
```

Foreign schema options

Option	Required	Default	Description
`import_default`	No	`false`	Whether to include column `DEFAULT` expressions in the foreign table definitions.
`import_not_null`	No	`true`	Whether to include column `NOT NULL` constraints in the foreign table definitions.
`keep_custom_types`	No	`false`	Sybase only. When `false` , user-defined types are resolved to their underlying system types. When `true` , the original user-defined types are preserved. If you enable this, make sure the equivalent domain types exist in Postgres before importing.

Using EXPLAIN

`EXPLAIN (VERBOSE)` shows the query issued on the remote system, along with cost-related parameters:

```
EXPLAIN (VERBOSE) SELECT * FROM mssql_table;
```

Configuring runtime variables

`tds_fdw` provides the following variables for debugging memory usage. Set them with the `SET` command:

```
SET tds_fdw.show_before_row_memory_stats = 1;
SET tds_fdw.show_after_row_memory_stats = 1;
SET tds_fdw.show_finished_memory_stats =
1;
```

Variable	Description
`tds_fdw.show_before_row_memory_stats`	Prints memory context stats to the PostgreSQL log before each row is fetched.
`tds_fdw.show_after_row_memory_stats`	Prints memory context stats to the PostgreSQL log after each row is fetched.
`tds_fdw.show_finished_memory_stats`	Prints memory context stats to the PostgreSQL log when a query finishes.

Troubleshooting

To get more detail from `tds_fdw` when diagnosing issues, increase the verbosity of console messages and set the message handler to `notice`:

```
SET client_min_messages TO
DEBUG3;
ALTER SERVER mssql_svr OPTIONS (SET msg_handler
'notice');
```

Then run your queries and review the full output. You can report issues on the GitHub's tds_fdw issues page.

Identifying the `tds_fdw` version

TDS foreign data wrapper allows you to identify the currently installed version of the extension by querying a system catalog. To check the version, connect to your Postgres server and enter:

```
SELECT extversion FROM pg_extension WHERE extname = 'tds_fdw';
```

output

```
 extversion
------------
 2.0.5
(1 row)
```