SQLdb Programming Reference 编辑

Databases portal

References:

- General info
- Libraries
- Field types
- Controls
- FAQ
- SQL how-to
- Working With TSQLQuery
- In-memory database applications

Tutorials/practical articles:

- Overview
- 0 - Database set-up
- 1 - Getting started
- 2 - Editing
- 3 - Queries
- 4 - Data modules
- SQLdb Programming Reference

Databases

Advantage - MySQL - MSSQL - Postgres - Interbase - Firebird - Oracle - ODBC - Paradox - SQLite - dBASE - MS Access - Zeos

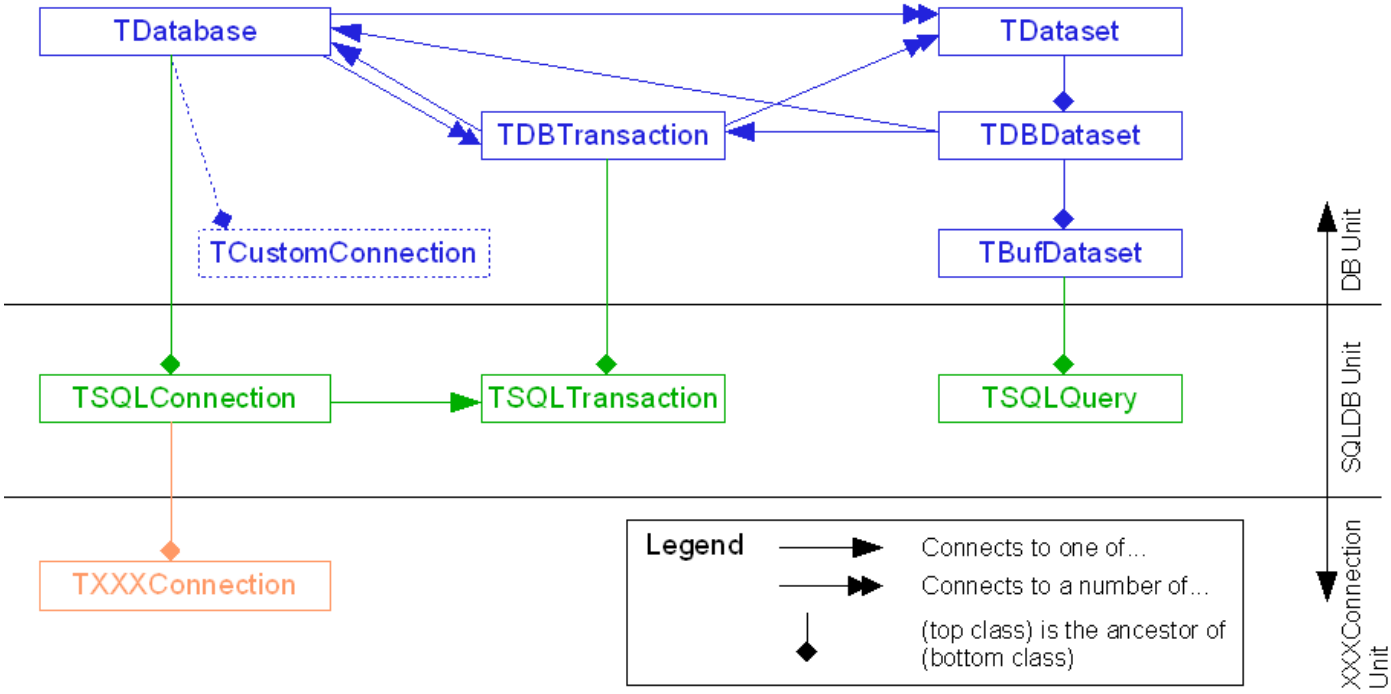
Documentation

Please see the official documentation at SQLDB documentation.

This article attempts to give some more detail about SQLDb; however the official documentation is authorative.

Class Structure

The following diagram attempts to show the hierarchy and required links of the MAIN components involved in SQLdb. It is certainly not exhaustive, nor does it use any "proper" diagram structure, so please don't try to read too much into it. I hope it will make it easier to work out which bits of the source code you need to look at to really work out what is happening.



Notes

- The link from TDatabase to TTransaction is Transactions, and is a list, implying many transactions are possible for the one database. However, a new link is defined from TSQLConnection to TSQLTransaction which is Transaction - a single transaction per database. This does not actually hide the previous link, but only the new link is published, and it is probably inadvisable to use the ancestor's link.
- Some of the inherited links need to be typecast to the new types to be useful. You can't call SQLQuery.Transaction.Commit, as Commit is only defined in TSQLTransaction. Call SQLTransaction.Commit, or "(SQLQuery.Transaction as TSQLTransaction).Commit"

Interaction

TConnection

Documentation: TSQLConnection documentation

A TConnection represents a connection to an SQL database. In daily use, you will use the descendent for a specific database (e.g. TIBConnection for Interbase/wiki/freepascal/Firebird), but it is possible to use TConnection if you are trying to write database factory/database independent applications (note: it's probably more advisable to use TSQLConnector). In this object, you specify connection-related items such as hostname, username and password. Finally, you can connect or disconnect (using the .Active or .Connected property)

Most database allow muliple concurrent connections from the same program/user.

TSQLTransaction

Documentation: TSQLTransaction

This object represents a transaction on the database. In practice, at least one transaction needs to be active for a database, even if you only use it for reading data. When using a single transaction, set the TConnection.Transaction property to the transaction to set the default transaction for the database; the corresponding TSQLTransaction.Database property should then automatically point to the connection.

Setting a TSQLTransaction to .Active/calling .StartTransaction starts a transaction; calling .Commit or .RollBack commits (saves) or rolls back (forgets/aborts) the transaction. You should surround your database transactions with these unless you use .Autocommit or CommitRetaining.

TSQLQuery

Documentation: TSQLQuery documentation

See Working With TSQLQuery for more details.

TSQLQuery is an object that embodies a dataset from a connection/transaction pair using its SQL property to determines what data is retrieved from the database into the dataset.

When working with it, you therefore need to at least specify the transaction, connection amd SQL properties. The TSQLQuery is an important part in the chain that links databound controls to the database. As said, the SQL property determines what SELECT query is run against the database to get data. FPC will try to determine what corresponding SQL INSERT, UPDATE and DELETE statements should be used in order to process user/program generated data changes. If necessary, the programmer can fine tune this and manually specify the InsertSQL, UpdateSQL and DeleteSQL properties.

DataSource

A TDataSource object keeps track of where in a dataset (such as TSQLQuery) data bound components are. The programmer should specify the corresponding TSQLQuery object for this to work.

Databound controls such as DBGrid

These controls must be linked to a DataSource. They will often have properties that indicate what fields in the DataSource they show.

Data modules

Data modules can be used to store non-visual components such as **T*Connection**, **TSQLTransaction**, **TSQLQuery** etc. Data modules also let you share components between forms.

See SQLdb Tutorial4.

分享到微信

分享到QQ

分享到微博