

List of drivers and TDS versions supported

Last updated by | Vitor Tomaz | Aug 5, 2020 at 12:34 PM PDT

Contents

- [TDS versions supported](#)
 - [List of supported TDS Drivers](#)
 - [Retry Logic](#)

TDS versions supported

Tabular Data Stream (TDS) is an application layer protocol, used to transfer data between a database server and a client. It was initially designed and developed by Sybase Inc. for their Sybase SQL Server relational database engine in 1984, and later by Microsoft in Microsoft SQL Server.

The TDS protocol supports the "TDS 7.0", "TDS 7.1", "TDS 7.2", "TDS 7.3", and "TDS 7.4" explicit dialects.

In SAWA v2, clients running TDS 7.4 or above will accept redirection and connect directly to the SQL Server Instance running on the tenant ring.

Clients running TDS lower than 7.4 will have to go thru a SNI proxy to connect to the SQL Engine instances running on the tenant rings.

For clients running TDS 7.4 or above, the data access provider redirects the connection transparently. To the client, a redirected connection appears to be a connection to the server instance identified by the initial Server Name.


Redirection occurs in the Login Response TDS token stream from Server to Client. The login response is a token stream consisting of information about the server's characteristics, optional information, and error messages, followed by a completion message.


The server can send, as part of the login response, one or more ENVCHANGE token data. ENVCHANGE Type 20 (Routing) is sent in response to route the client to an alternate server. The ENVCHANGE stream returns routing information for the alternate server.

List of supported TDS Drivers




Any TDS 7.0", "TDS 7.1", "TDS 7.2", "TDS 7.3", and "TDS 7.4" explicit dialects.

List of supported drivers for redirection

Name	Release Mechanism	Release Versions (TDS versions)
ADO.NET  SqlClient	.NET Framework	4.5 Blue/4.5.1 (7.4), 4.5 RTM (7.4), 4.0 (7.3a), 3.5
Microsoft ODBC Driver 11 for SQL Server on Windows (former SNAC)	SQL Server	11 (7.4)
JDBC Driver	OOB	4.0 (7.4)
PHP Driver	Open Source	Based on SQL ODBC Driver (7.4)
Node.JS	Open Source, Zumo	0.8.9 x86 (as on 20150731)
Microsoft ODBC Driver 11 for SQL Server on Linux	OOB	11 (7.4)


Node.js: <https://azure.microsoft.com/en-us/documentation/articles/sql-database-develop-nodejs-simple-windows/> 

Downloads

[Download the Microsoft ODBC Driver 11 for SQL Server on Windows](#)  [Download Command line Utilities for Microsoft ODBC Driver 11 for SQL Server](#)  [Download Microsoft ODBC Driver 11 for SQL Server on Red Hat Linux](#) 

Retry Logic


Sterling DB follows a principle of “fail fast” instead of attempting to retry on behalf of the client.

Customers will need to ensure that they have proper retry logic in place to handle the increased rate of error (any retry logic that worked with Production Azure SQL DB should work - such as [The Transient Fault Handling Application Block](#) ).


Sterling is attempting to make all transient errors of database unavailability a single error number – 40613 – database unavailable

FAQs

Q: Should we use ID= username@servername in the connection string

A: It will NOT hurt if somebody created a connection string with username@servername however, it is mostly unnecessary these days (due to new drivers) and it will break Elastic Scale. ([See known issues page](#) 

Q: What exactly is the SSL standard being used for connections to an azure SQL database?

A: Same as Sql Server 2016 as outlined at <https://support.microsoft.com/en-us/help/3135244/tls-1-2-support-for-microsoft-sql-server> . TLS 1.2 and SSL 3.0

Q: When connecting from [ADO.NET](#) to a database in Azure SQL, what are the protocols and ciphers used (if there's multiple options, please have them list those) to establish the connection and to keep an encrypted tunnel?

A: <https://azure.microsoft.com/en-us/documentation/articles/sql-database-security/#compliance> has details. If they have more specific questions after reading then we can spend more time.

Q: Any suggested driver versions?

A: The latest always has the known issues/bugs fixed and typically has better reliability.

For .NET framework, latest is currently .NET Framework 4.5.2. We have added more diagnostic information to 4.5.2 for SQL DB v12.

Q: Any suggestions for retry logic?

A: As for your retry logic, you should back off minimum five seconds before you retry to open a connection.

I would actually recommend six seconds. 😊

Also, for retry lock, we recommend that you retry the entire logic of { open connection, execute command} even if you get transient error on executing command.

This ensures that connections are returned to the pool and cleaned up properly.

<MSInternal 20150624>

On connection pool, if we failed to open a physical connection (vs. getting opened connection from the pool), we will cache the exception and rethrow it next 5s.

After 5s, it will try to open a physical connection again and if it fails again it will rethrow the cached error for next 10s.

It can block up to 1min.

We recommend our customers to back off min 5s so that there is less confusion about the cached error vs. new error.

</MSInternal 20150624>

Q: Sometimes i need to stop and re-start my application (and not just retry connection)?

A: Can get around the issue by creating new pool by using different app id on the connection string.

OR Another option is for you to clear pool. Try not to use clearallpool. But clear the specific pool.

Problem is detecting the problem programmatically. If you are hitting this issue, you will see consistent timeout issue on command execution.

<MSInternal 20150624>

We have an issue where we don't properly repair connection between our FE and BE when the database fails over. When connection is pulled from the connection pool, it sends reset to our service and it suppose to repair the connection if it is broken between FE and BE.

We will be fixing it, but meantime, you can get around the issue by creating new pool by using different app id on the connection string.

</MSInternal 20150624>

Q: Can Azure tell the difference between one computer behind NAT and not behind NAT?

A: No, we cannot tell the difference.

How good have you found this content?

