# sqlcmd SQL Server shell

## CONCEPTS

Typed lines collect in a batch; nothing runs until GO on a line of its own. GO and :commands belong to sqlcmd, not T-SQL, and must start the line.

**Connect** -S -U -E -d
↓ type / :r
**Batch** lines, $(var) expanded
↓ GO
**Server** runs the batch
↓ results
**Output** screen, -o, :out

Two variants share the name: go-sqlcmd (standalone, has sqlcmd --version and create) and ODBC sqlcmd from mssql-tools18. sqlcmd -? shows the version on both.

ODBC sqlcmd 18 encrypts by default (-Nm) and validates the certificate, so a local server with a self-signed cert needs -C.

## CONNECT

`sqlcmd -S <host>,<port> -U <user>`
Connect with a SQL login

`sqlcmd -S <host>\<instance>`
Connect to a named instance

`sqlcmd -E`
Use integrated (Windows/ Kerberos) auth

`SQLCMDPASSWORD=<pw> sqlcmd -U <user>`
Pass the password via environment

`sqlcmd -d <db>`
Start in a database

`sqlcmd -C`
Trust the server certificate

`sqlcmd -N<s|m|o>`
Encryption: strict, mandatory, optional

`sqlcmd -G -S <srv>.database.windows.net`
Sign in with Microsoft Entra ID

`sqlcmd -l <seconds>`
Login timeout (default 8)

`:connect <host> -U <user>`
Switch server inside a session

## RUN SQL

`sqlcmd -Q "<sql>"`
Run a query and exit

`sqlcmd -i <file.sql>`
Run a script file

`sqlcmd -i <in.sql> -o <out.txt>`
Write all output to a file

`GO`
Run the buffered batch

`:r <file>`
Read a file into the batch

`:reset`
Discard the current batch

`sqlcmd -t <seconds>`
Query timeout (default none)

`:quit`
Leave sqlcmd

## VARIABLES

`:setvar <var> "<value>"`
Define a scripting variable

`$(<var>)`
Insert a variable's value

`sqlcmd -v <var>="<value>"`
Set a variable from the shell

**Built-in variables**

| Variable | Set by |
|---|---|
| SQLCMDSERVER | -S, :connect |
| SQLCMDUSER | -U |
| SQLCMDPASSWORD | -P |
| SQLCMDDBNAME | -d |
| SQLCMDINI | startup script path |
| SQLCMDEDITOR | editor for :ed |

## OUTPUT

`sqlcmd -W`
Trim trailing spaces from columns

`sqlcmd -s "<sep>"`
Set the column separator

`sqlcmd -h -1`
Omit column headers

`sqlcmd -y <n>`
Max width of (max)/xml values

`SET NOCOUNT ON;`
Hide "(n rows affected)"

`:out <file>`
Send results to a file

`sqlcmd --vertical`
One column per line (go-sqlcmd)

## ERRORS & EXIT CODES

`sqlcmd -b`
Stop and exit 1 on an error

`:on error exit`
Stop the script on an error

`sqlcmd -V <severity>`
Minimum severity that sets the exit code

`:exit(<query>)`
Exit with a query result as code

## GO-SQLCMD CONTAINERS

`sqlcmd create mssql --accept-eula`
Start SQL Server in a container

`sqlcmd create mssql --accept-eula --using <url.bak>`
Same, restoring a backup

`sqlcmd create mssql --tag <tag>`
Pick a SQL Server version

`sqlcmd query "<sql>"`
Query the current context

`sqlcmd config use-context <name>`
Switch context

`sqlcmd config connection-strings`
ODBC, ADO.NET, JDBC strings

`sqlcmd stop / sqlcmd start`
Stop or start the container

`sqlcmd delete`
Remove container and context

## T-SQL HELPERS

`SELECT name FROM sys.databases;`
List databases

`USE <db>;`
Switch database

`SELECT name FROM sys.tables;`
List tables in this database

`EXEC sp_help '<table>';`
Describe a table

`SELECT TOP (10) * FROM <table>;`
First rows (no LIMIT in T-SQL)

`SELECT @@VERSION;`
Server version