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 -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

`: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

RUN SQL (cont.)

`GO`
Run the buffered batch

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

`: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

OUTPUT

`sqlcmd -W`
Trim trailing spaces from columns

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

`sqlcmd -h -1`
Omit column headers

ERRORS & EXIT CODES

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

GO-SQLCMD CONTAINERS

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

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