

Versão

SQL Server 2022

Filtrar por título

Visão geral do sqlcmd

Visão geral do go-sqlcmd

Iniciar o Utilitário

Usar o Utilitário

Conectar-se ao Mecanismo de Banco de

Dados

Executar arquivos de script T-SQL

Usar com variáveis de script

Editar scripts SQLCMD com o Editor de

Consultas

Baixar PDF

> SQL Server Management Studio (SSMS)

> SqlPackage

> SQL Server Profiler

> Auxiliares nativos do Visual Studio

> Recursos estendidos

Baixar PDF

Learn / SQL /

utilitário sqlcmd

Artigo • 12/12/2023 • 5 colaboradores

Comentários

Escolha o shell de comando

Bash PowerShell **Cmd**

Neste artigo

[Descobrir qual versão foi instalada](#)

[Baixar e instalar o sqlcmd](#)

[Pré-instalado](#)

[Sintaxe](#)

[Mostrar mais 7](#)

Aplica-se a: [SQL Server](#) [Banco de Dados SQL do Azure](#) [Instância Gerenciada de SQL do Azure](#) [Azure Synapse Analytics](#) [PDW \(Analytics Platform System\)](#)

O utilitário do `sqlcmd` permite que você insira as instruções Transact-SQL, os procedimentos do sistema e os arquivos de script usando vários modos:

- No prompt de comando.
- No **Editor de Consulta** no modo SQLCMD.
- Em um arquivo de script do Windows.
- Em uma etapa de trabalho do sistema operacional (`cmd.exe`) de um trabalho do SQL Server Agent.

Descobrir qual versão foi instalada

Há duas versões do `sqlcmd`:

- O `sqlcmd` baseado em `go-mssqldb`, às vezes chamado de `go-sqlcmd`. Esta versão é uma ferramenta autônoma que você pode baixar independentemente do SQL Server.
- O `sqlcmd` baseado em ODBC, disponível com o SQL Server ou os Utilitários de Linha de Comando da Microsoft e parte do pacote `mssql-tools` no Linux.

Para determinar a versão instalada, execute a seguinte instrução na linha de comando:

```
Prompt de comando do Windows Copiar
sqlcmd -?
```

sqlcmd (Go) sqlcmd (ODBC)

Se você estiver usando a nova versão do `sqlcmd` (Go), a saída será semelhante ao exemplo a seguir:

```
Saída Copiar
Version: 1.3.1
```

Verificar a versão

Você pode usar o `sqlcmd --version` para determinar qual versão está instalada. Você deve ter pelo menos a versão 1.0.0 instalada.

Observação

No SQL Server 2014 (12.x) e versões anteriores, confira o utilitário `sqlcmd`.

Para usar o `sqlcmd` no Linux, confira [Instalar o sqlcmd e o bcp no Linux](#).

Importante

A instalação do `sqlcmd` (Go) por meio de um gerenciador de pacotes substituirá o `sqlcmd` (ODBC) pelo `sqlcmd` (Go) no caminho do seu ambiente. Todas as sessões de linha de comando

Recursos adicionais

Eventos

Desafio de habilidades de nuvem

15 de nov., 14 - 15 de jan., 14

Escolha entre 12 coleções exclusivas no Microsoft Learn. Insira para ganhar um passe de evento VIP para o próximo Microsoft Ignite ou Microsoft Build!

[Registrar agora](#)

Documentação

sqlcmd – Usar o utilitário sqlcmd - SQL Server

Saiba como usar o utilitário `sqlcmd` para execução interativa ad hoc de scripts e instruções Transact-SQL e automatize tarefas de script Transact-SQL.

Utilitário sqlcmd: iniciar o utilitário sqlcmd - SQL Server

Saiba como iniciar o utilitário `sqlcmd`, que permite inserir instruções Transact-SQL, procedimentos de sistema e arquivos de script, no modo SQLCMD ou...

Utilitário sqlcmd: conectar-se ao mecanismo de banco de dados com sqlcmd - SQL Server

Saiba como selecionar o protocolo usado pelo `sqlcmd` para se comunicar com o SQL Server. As opções são: TCP/IP, pipes nomeados e memória compartilhada.

[Mostrar mais 5](#)

atuais precisarão ser recriadas e reabertas para que essa mudança entre em vigor. O `sqlcmd` (ODBC) não será removido e ainda poderá ser usado especificando o caminho completo para o executável. Você também pode atualizar sua variável `PATH` para indicar qual terá precedência. Para fazer isso no Windows 11, abra **Configurações do sistema** e acesse **Sobre > Configurações avançadas do sistema**. Quando as **Propriedades do Sistema** forem abertas, selecione o botão **Variáveis de Ambiente**. Na metade inferior, em **Variáveis do Sistema**, selecione **Caminho** e, em seguida, selecione **Editar**. Se o local no qual o `sqlcmd` (Go) for salvo (`C:\Program Files\sqlcmd` é o padrão) estiver listado antes de `C:\Program Files\Microsoft SQL Server\<version>\Tools\Binn`, o `sqlcmd` (Go) será usado. Você pode inverter a ordem para tornar o `sqlcmd` (ODBC) o padrão novamente.

Baixar e instalar o sqlcmd

sqlcmd (Go) sqlcmd (ODBC)

O `sqlcmd` (Go) pode ser instalado em várias plataformas, no Microsoft Windows, macOS e Linux.

Windows macOS Linux

apt (Debian/Ubuntu)

1. Importe as chaves GPG do repositório público.

```
Bash Copiar
curl https://packages.microsoft.com/keys/microsoft.asc | sudo tee /etc/apt/tru
```

2. Adicione o repositório da Microsoft, no qual o segmento `ubuntu/20.04` pode estar `debian/11`, `ubuntu/20.04` ou `ubuntu/22.04`.

```
Bash Copiar
add-apt-repository "$(wget -qO- https://packages.microsoft.com/config/ubuntu/2
```

3. Instale o `sqlcmd` (Go) com `apt`.

```
Bash Copiar
apt-get update
apt-get install sqlcmd
```

yum (Fedora/CentOS)

1. Importe a chave do repositório da Microsoft.

```
Bash Copiar
rpm --import https://packages.microsoft.com/keys/microsoft.asc
```

2. Baixe o arquivo de configuração do repositório, no qual o segmento `centos/8` pode estar `centos/8`, `fedora/32`, `opensuse/42.3`, `rhel/8` ou `sles/15`. Se a versão do sistema operacional não corresponder diretamente a uma dessas opções, provavelmente você poderá usar com êxito um arquivo de configuração de repositório de uma versão. Por exemplo, `centos/8` pode ser usado em um ambiente que executa o CentOS 7.

```
Bash Copiar
curl -o /etc/yum/repos.d/packages-microsoft-com-prod.repo https://packages.mic
```

3. Instale `sqlcmd` (Go) com `yum`.

```
Bash Copiar
yum install sqlcmd
```

Download direto

1. Baixe o ativo `-linux-x64.tar.bz2` ou `-linux-arm.tar.bz2` correspondente da versão [mais recente](#) do `sqlcmd` (Go) no repositório de código do GitHub.
2. Extraia o arquivo `sqlcmd` da pasta zip baixada.

Pré-instalado

Azure Cloud Shell

Você pode experimentar o utilitário `sqlcmd` do Azure Cloud Shell, que vem pré-instalado por padrão:

- [Iniciar o Cloud Shell](#)

Azure Data Studio

Para executar instruções SQLCMD no [Azure Data Studio](#), selecione **Habilitar SQLCMD** na barra de ferramentas do editor.

SQL Server Management Studio (SSMS)

Para executar instruções SQLCMD no [SQL Server Management Studio](#) (SSMS), selecione Modo SQLCMD na lista suspensa do menu Consultar na parte superior da navegação.

O SSMS usa o Microsoft .NET Framework `sqlclient` para execução nos modos regular e SQLCMD no **Editor de Consultas**. Quando o `sqlcmd` é executado na linha de comando, o `sqlcmd` usa o driver ODBC. Devido às diferentes opções padrão que podem ser aplicadas, é possível observar um comportamento diferente ao executar a mesma consulta no no Modo SQLCMD e no utilitário `sqlcmd`.

Sintaxe

sqlcmd (Go) sqlcmd (ODBC)

```
Saída Copiar

Usage:
  sqlcmd [flags]
  sqlcmd [command]

Examples:
# Install/Create, Query, Uninstall SQL Server
sqlcmd create mssql --accept-eula --using https://aka.ms/AdventureworksLT.bak
sqlcmd open ads
sqlcmd query "SELECT @@version"
sqlcmd delete
# View configuration information and connection strings
sqlcmd config view
sqlcmd config cs

Available Commands:
  completion  Generate the autocompletion script for the specified shell
  config      Modify sqlconfig files using subcommands like "sqlcmd config use-context
  create      Install/Create SQL Server, Azure SQL, and Tools
  delete      Uninstall/Delete the current context
  help        Help about any command
  open        Open tools (e.g ADS) for current context
  query       Run a query against the current context
  start       Start current context
  stop        Stop current context

Flags:
  -?, --?          help for backwards compatibility flags (-s, -U, -E etc.)
  -h, --help       help for sqlcmd
  --sqlconfig string configuration file (default "/Users/<currentUser>/sqlcmd/sql
  --verbosity int  log level, error=0, warn=1, info=2, debug=3, trace=4 (default
  --version        print version of sqlcmd

Use "sqlcmd [command] --help" for more information about a command.
```

Para obter informações mais detalhadas sobre a sintaxe e o uso do `sqlcmd`, confira [sintaxe do sqlcmd ODBC](#).

Alterações interruptivas do sqlcmd (ODBC)

Várias opções e comportamentos são alterados no utilitário `sqlcmd` (Go). Para obter a lista mais atualizada de sinalizadores ausentes para compatibilidade com versões anteriores, visite a discussão [Priorizar a implementação de sinalizadores de back-compat](#) no GitHub.

- Em versões anteriores do `sqlcmd` (Go), a opção `-p` foi removida temporariamente, e as senhas para a Autenticação do SQL Server só podiam ser fornecidas por meio destes mecanismos:
 - A variável de ambiente `SQLCMDPASSWORD`
 - O comando `:CONNECT`
 - Quando solicitado, o usuário podia digitar a senha para concluir uma conexão
- `-r` requer um argumento `@` ou `!`
- A opção `-R` foi removida.
- A opção `-I` foi removida. Para desabilitar o comportamento do identificador delimitado, adicione `SET QUOTED IDENTIFIER OFF` aos seus scripts.
- `-N` agora usa um valor de cadeia de caracteres que pode ser de `true`, `false` ou `disable` para especificar a opção de criptografia. (`default` é o mesmo que omitir o parâmetro)

- o Se `-N` e `-C` não forem fornecidos, o `sqlcmd` negociará a autenticação com o servidor sem validar o certificado do servidor.
 - o Se `-N` for fornecido, mas `-C` não for, o `sqlcmd` exigirá a validação do certificado do servidor. Um valor `false` para criptografia ainda pode levar à criptografia do pacote de logon.
 - o Se forem fornecidos `-N` e `-C`, o `sqlcmd` usará seus valores para negociação de criptografia.
 - o Mais informações sobre a negociação de criptografia de cliente/servidor podem ser encontradas em [MS-TDS PRELOGIN](#).
- `-u` O arquivo de saída Unicode gerado tem o BOM (marca de ordem de byte) Little-Endian UTF-16 gravado nele.
 - Alguns comportamentos que foram mantidos para manter a compatibilidade com `osql` podem ser alterados, como o alinhamento de cabeçalhos de coluna para alguns tipos de dados.
 - Todos os comandos devem caber em uma linha, mesmo `EXIT`. O modo interativo não verifica se há parênteses ou aspas abertos para comandos e não solicita linhas sucessivas. Esse comportamento é diferente da versão ODBC, que permite que a consulta executada por `EXIT(query)` abra várias linhas.

As conexões do utilitário `sqlcmd` (Go) são limitadas a conexões TCP. No momento, não há suporte para pipes nomeados no driver `go-mssqldb`.

Aprimoramentos

- `:connect` agora tem um parâmetro opcional `-G` para selecionar um dos métodos de autenticação do Banco de Dados SQL do Azure – `SqlAuthentication`, `ActiveDirectoryDefault`, `ActiveDirectoryIntegrated`, `ActiveDirectoryServicePrincipal`, `ActiveDirectoryManagedIdentity`, `ActiveDirectoryPassword`. Para obter mais informações, consulte [Autenticação do Azure Active Directory](#). Se `-G` não for fornecido, será usada a segurança integrada ou a autenticação do SQL Server, dependendo da presença do parâmetro de nome de usuário `-U`.
- O novo parâmetro de linha de comando `--driver-logging-level` permite que você veja os rastreamentos no driver `go-mssqldb`. Use `64` para ver todos os rastreamentos.
- Agora o `sqlcmd` pode imprimir resultados usando o formato vertical. Use a nova opção de linha de comando `-F vertical` para defini-lo. A variável de script `SQLCMDFORMAT` também a controla.

Opções de linha de comando

Opções relacionadas a logon

-A

Entre no SQL Server com uma DAC (conexão de administrador dedicada). Esse tipo de conexão é usado para solucionar um problema no servidor. Essa conexão funciona apenas com computadores servidor compatíveis com DAC. Se a DAC não estiver disponível, o `sqlcmd` gerará uma mensagem de erro e será encerrado. Para obter mais informações sobre a DAC, consulte [Conexão de diagnóstico para administradores de banco de dados](#). A opção `-A` não é compatível com a opção `-G`. Ao se conectar ao Banco de Dados SQL do Azure usando `-A`, você precisa ser um administrador no SQL Server lógico. A DAC não está disponível para um administrador do Azure AD.

-C

Essa opção é usada pelo cliente para configurá-la para confiar implicitamente no certificado do servidor sem validação. Essa opção é equivalente à opção `TRUSTSERVERCERTIFICATE = true` do ADO.NET.

Para o utilitário `sqlcmd` (Go), as seguintes condições também se aplicam:

- Se `-N` e `-C` não forem fornecidos, o `sqlcmd` negociará a autenticação com o servidor sem validar o certificado do servidor.
- Se `-N` for fornecido, mas `-C` não for, o `sqlcmd` exigirá a validação do certificado do servidor. Um valor `false` para criptografia ainda pode levar à criptografia do pacote de logon.
- Se forem fornecidos `-N` e `-C`, o `sqlcmd` usará seus valores para negociação de criptografia.

-d db_name

Emite uma instrução `USE <db_name>` ao iniciar o `sqlcmd`. Essa opção define a variável de script do `sqlcmd` `SQLCMDBNAME`. Esse parâmetro especifica o banco de dados inicial. O padrão é a propriedade do banco de dados padrão de seu logon. Se o banco de dados não existir, será gerada uma mensagem de erro e o `sqlcmd` será encerrado.

-d

Interpreta o nome do servidor fornecido para `-S` como um DSN em vez de um nome de host. Para obter mais informações, confira *Suporte para DSN no sqlcmd e no bcp* em [Conectar-se com sqlcmd](#).

🕒 Observação

A opção `-D` só está disponível em clientes Linux e macOS. Em clientes Windows, ela fazia referência a uma opção agora obsoleta que foi removida e ignorada.

-l *login_timeout*

Especifica o número de segundos antes que um logon do `sqlcmd` no driver ODBC expire quando você tentar se conectar a um servidor. Essa opção define a variável de script do `sqlcmd` `SQLCMDLOGINTIMEOUT`. O tempo limite padrão de logon do `sqlcmd` é de oito segundos. Ao usar a opção `-G` para se conectar ao Banco de Dados SQL do Azure ou ao Azure Synapse Analytics e autenticar-se usando o Azure AD, é recomendado estipular um valor de tempo limite de pelo menos 30 segundos. O tempo limite do logon precisa ser um número entre `0` e `65534`. Se o valor fornecido não for numérico ou não estiver nesse intervalo, o `sqlcmd` vai gerar uma mensagem de erro. Um valor de `0` especifica o tempo limite como infinito.

-E

Usa uma conexão confiável em vez de um nome de usuário e uma senha para entrar no SQL Server. Por padrão, sem a especificação de `-E`, o `sqlcmd` usa a opção de conexão confiável.

A opção `-E` ignora possíveis definições de variável de ambiente de nome de usuário e senha como `SQLCMDPASSWORD`. Se a opção `-E` for usada com a opção `-U` ou `-P`, uma mensagem de erro será gerada.

-g

Define a configuração de Criptografia de Coluna como `Enabled`. Para obter mais informações, consulte [Always Encrypted](#). Há suporte apenas para as chaves mestras armazenadas no Repositório de Certificados do Windows. A opção `-g` exige, no mínimo, o `sqlcmd` versão [13.1](#). Para determinar a versão, execute `sqlcmd -?`.

\- G

Essa opção é usada pelo cliente ao se conectar ao Banco de Dados SQL do Azure ou o Azure Synapse Analytics para especificar que o usuário seja autenticado usando a autenticação do Azure AD. Essa opção define a variável de script do `sqlcmd` `SQLCMDUSEAAD = true`. A opção `-G` exige, no mínimo, o `sqlcmd` versão [13.1](#). Para determinar a versão, execute `sqlcmd -?`. Para obter mais informações, confira [Conectando-se ao Banco de Dados SQL ou ao Azure Synapse Analytics usando a Autenticação do Azure Active Directory](#). A opção `-A` não é compatível com a opção `-G`.

A opção `-G` só se aplica ao Banco de Dados SQL do Azure e ao Azure Synapse Analytics.

No momento, a autenticação integrada do Azure AD não tem suporte no Linux nem no macOS. A autenticação integrada do Azure AD requer o [Microsoft ODBC Driver 17 for SQL Server](#) versão 17.6.1 ou posterior e um ambiente Kerberos configurado corretamente.

Para obter mais informações sobre a autenticação do Azure Active Directory, consulte [Autenticação do Azure Active Directory no sqlcmd](#).

-H *workstation_name*

Um nome de estação de trabalho. Essa opção define a variável de script do `sqlcmd` `SQLCMDWORKSTATION`. O nome da estação de trabalho é listado na coluna `hostname` da exibição de catálogo `sys.sysprocesses` e pode ser retornado usando o procedimento armazenado `sp_who`. Se essa opção não for especificada, o padrão será o nome do computador atual. Esse nome pode ser usado para identificar diferentes sessões do `sqlcmd`.

-j

Imprime mensagens de erro bruto na tela.

-K *application_intent*

Declara o tipo de carga de trabalho de aplicativo ao conectar-se a um servidor. O único valor com suporte no momento é `ReadOnly`. Se `-K` não for especificado, o `sqlcmd` não dará suporte à conectividade com uma réplica secundária em um grupo de disponibilidade. Para obter mais informações, confira [Réplicas secundárias ativas: réplica secundária para leitura \(Grupos de Disponibilidade Always On\)](#).

-M multisetnet_failover

Sempre especifique `-M` ao se conectar ao ouvinte de um grupo de disponibilidade do SQL Server ou a uma Instância de cluster de failover do SQL Server. `-M` proporciona a detecção mais rápida e a conexão com o servidor (atualmente) ativo. Se `-M` não estiver especificado, `-M` estará desativado. Para saber mais, veja [Ouvintes, Conectividade do Cliente e Failover do Aplicativo, Criação e Configuração de Grupos de Disponibilidade \(SQL Server\)](#), [Clustering de Failover e Grupos de Disponibilidade AlwaysOn \(SQL Server\)](#) e [Secundárias Ativas: Réplicas Secundárias Legíveis \(Grupos de Disponibilidade AlwaysOn\)](#).

-n

Essa opção é usada pelo cliente para solicitar uma conexão criptografada.

Para o utilitário `sqlcmd` (Go), o `-n` agora usa um valor de cadeia de caracteres que pode ser de `true`, `false` ou `disable` para especificar a opção de criptografia. (`default` é o mesmo que omitir o parâmetro):

- Se `-n` e `-c` não forem fornecidos, o `sqlcmd` negociará a autenticação com o servidor sem validar o certificado do servidor.
- Se `-n` for fornecido, mas `-c` não for, o `sqlcmd` exigirá a validação do certificado do servidor. Um valor `false` para criptografia ainda pode levar à criptografia do pacote de logon.
- Se forem fornecidos `-n` e `-c`, o `sqlcmd` usará seus valores para negociação de criptografia.

-P password

Uma senha especificada pelo usuário. As senhas diferenciam maiúsculas de minúsculas. Se a opção `-u` for usada, a opção `-p` não for usada e a variável de ambiente `SQLCMDPASSWORD` não tiver sido definida, o `sqlcmd` solicitará uma senha ao usuário. Não recomendamos o uso da senha nula (em branco), mas você pode especificar a senha nula usando um par de aspas duplas contíguas para o valor do parâmetro (`""`).

📌 Importante

O uso de `-p` deve ser considerado inseguro. Evite fornecer a senha na linha de comando. Como alternativa, use a variável de ambiente `SQLCMDPASSWORD` ou insira de modo interativo a senha omitindo a opção `-p`.

Recomendamos usar uma [senha forte](#).

A solicitação de senha é exibida no console, como a seguir: `Password:`

A entrada do usuário está oculta. Isso significa que nada é exibido e o cursor fica em posição.

A variável de ambiente `SQLCMDPASSWORD` permite que você defina uma senha padrão para a sessão atual. Assim, senhas não precisam ser embutidas em código nos arquivos em lote. O exemplo a seguir define primeiro a variável `SQLCMDPASSWORD` no prompt de comando e acessa o utilitário `sqlcmd`.

No prompt de comando, digite:

```
Prompt de comando do Windows 📄 Copiar  
  
SET SQLCMDPASSWORD=p@a$$w0rd  
sqlcmd
```

Se a combinação de nome de usuário e senha estiver incorreta, uma mensagem de erro será gerada.

📌 Observação

A variável de ambiente `OSQLPASSWORD` foi mantida para compatibilidade com versões anteriores. A variável de ambiente `SQLCMDPASSWORD` tem precedência em relação à variável de ambiente `OSQLPASSWORD`. Isso significa que o `sqlcmd` e `osql` podem ser usados próximos um do outro sem interferência. Os scripts antigos continuarão funcionando.

Será gerada uma mensagem de erro se a opção `-p` for usada com a opção `-E`.

Será gerada uma mensagem de erro se a opção `-p` for acompanhada de mais de um argumento e o programa será encerrado.

Uma senha contendo caracteres especiais pode gerar uma mensagem de erro. Você deve fazer o escape de caracteres especiais ao usar `-p` ou usar a variável de ambiente `SQLCMDPASSWORD` como alternativa.

-S [protocol:]server[\instance_name][,port]

Especifica a instância do SQL Server à qual você deseja se conectar. Ele define a variável de script do `sqlcmd` `SQLCMDSERVER`.

Especifique `server_name` para conectar-se à instância padrão do SQL Server nesse computador servidor. Especifique `server_name[\instance_name]` para conectar-se a uma instância nomeada do SQL Server nesse computador do servidor. Se nenhum servidor for especificado, o `sqlcmd` se conectará à

instância padrão do SQL Server no computador local. Essa opção é necessária quando você executa o `sqlcmd` em um computador remoto na rede.

`Protocolo` pode ser `tcp` (TCP/IP), `ipc` (memória compartilhada) ou `np` (pipes nomeados).

Se você não especificar um `server_name[instance_name]` ao iniciar o `sqlcmd`, o SQL Server verifica se há uma variável de ambiente `SQLCMDSERVER` e usa-a.

🕒 Observação

A variável de ambiente `OSQLSERVER` foi mantida para compatibilidade com versões anteriores. A variável de ambiente `SQLCMDSERVER` tem precedência em relação à variável de ambiente `OSQLSERVER`. Isso significa que o `sqlcmd` e `osql` podem ser usados próximos um do outro sem interferência. Os scripts antigos continuarão funcionando.

-U login_id

O nome de logon ou o nome de usuário de banco de dados independente. Para usuários de banco de dados independente, é necessário fornecer a opção de nome de banco de dados (`-d`).

🕒 Observação

A variável de ambiente `OSQLUSER` foi mantida para compatibilidade com versões anteriores. A variável de ambiente `SQLCMDUSER` tem precedência em relação à variável de ambiente `OSQLUSER`. Isso significa que o `sqlcmd` e `osql` podem ser usados próximos um do outro sem interferência. Os scripts antigos continuarão funcionando.

Se as opções `-U` ou `-P` não forem especificadas, o `sqlcmd` tentará se conectar usando o modo de Autenticação do Windows. A autenticação se baseia na conta do Windows do usuário que está executando o `sqlcmd`.

Se a opção `-U` for usada com a opção `-E` (descrita adiante neste artigo), uma mensagem de erro será gerada. Será gerada uma mensagem de erro se a opção `-U` for acompanhada de mais de um argumento e o programa será encerrado.

-z new_password

Altere a senha:

```
Prompt de comando do Windows Copiar
sqlcmd -U someuser -P somepassword -z a_new_password
```

-Z new_password

Altere a senha e saia:

```
Prompt de comando do Windows Copiar
sqlcmd -U someuser -P somepassword -Z a_new_password
```

Opções de entrada/saída

-f codepage | i:codepage[o:codepage] | o:codepage[i:codepage]

Especifica as páginas de código de entrada e saída. O número da página de código é um valor numérico que especifica um código de página instalada do Windows.

Regras de conversão de página de código:

- Se nenhuma página de código for especificada, o `sqlcmd` usará a página de código atual para arquivos de entrada e de saída, a menos que o arquivo de entrada seja um arquivo Unicode, para o qual nenhuma conversão é necessária.
- O `sqlcmd` reconhece arquivos de entrada Unicode big endian e little endian automaticamente. Se a opção `-u` for especificada, a saída será sempre o Unicode little-endian.
- Se nenhum arquivo de saída for especificado, a página de código de saída será a página de código de console. Essa abordagem habilita a saída a ser exibida corretamente no console.
- Assume-se que arquivos de entrada múltiplos tenham a mesma página de código. Arquivos de entrada Unicode e não Unicode podem ser misturados.

Insira `chcp` no prompt de comando para verificar a página de código de `cmd.exe`.

-i input_file[,input_file2...]

Identifica o arquivo que contém um lote de instruções Transact-SQL ou procedimentos armazenados. Poderão ser especificados vários arquivos que serão lidos e processados em ordem. Não use espaço

entre nomes de arquivos. O `sqlcmd` verificará primeiro se todos os arquivos especificados existem. Se um ou mais arquivos não existirem, o `sqlcmd` será encerrado. As opções `-i` e `-Q/-q` são mutuamente exclusivas.

Exemplos de caminho:

```
Saída Copiar
-i C:\<filename>
-i \\<Server>\<Share$>\<filename>
-i "C:\Some Folder\<file name>"
```

Os nomes de caminhos que contêm espaços deverão ficar entre aspas.

Esta opção pode ser usada mais de uma vez:

```
Prompt de comando do Windows Copiar
sqlcmd -i <input_file1> -i <input_file2>
```

-o *output_file*

Identifica o arquivo que recebe a saída do `sqlcmd`.

Se `-u` for especificado, *output_file* será armazenado no formato Unicode. Se o nome do arquivo não for válido, uma mensagem de erro será gerada e o `sqlcmd` será encerrado. O `sqlcmd` não dá suporte à gravação simultânea de vários processos `sqlcmd` no mesmo arquivo. A saída de arquivo está corrompida ou é incorreta. A opção `-f` também é relevante para formatos de arquivo. Esse arquivo será criado se não existir. Um arquivo com o mesmo nome de uma sessão `sqlcmd` anterior será substituído. O arquivo aqui especificado não é o arquivo `stdout`. Se um arquivo `stdout` for especificado, esse arquivo não será usado.

Exemplos de caminho:

```
Saída Copiar
-o C:< filename>
-o \\<Server>\<Share$>\<filename>
-o "C:\Some Folder\<file name>"
```

Os nomes de caminhos que contêm espaços deverão ficar entre aspas.

-r[0 | 1]

Redireciona a saída da mensagem de erro para a tela (`stderr`). Se você não especificar um parâmetro ou se especificar `0`, serão redirecionadas somente mensagens de erro com nível de severidade 11 ou acima disso. Se você especificar `1`, serão redirecionadas todas as saídas de mensagens de erro inclusive `PRINT`. Essa opção não terá efeito se você usar `-o`. Por padrão, mensagens são enviadas para `stdout`.

🕒 Observação

Para o utilitário `sqlcmd` (Go), `-r` requer um argumento `0` ou `1`.

-R

Aplicável a: utilitário `sqlcmd` ODBC.

Faz com que o `sqlcmd` localize colunas numéricas, de moeda, de data e de hora recuperadas do SQL Server com base na localidade do cliente. Por padrão, essas colunas são exibidas usando as configurações regionais do servidor.

-U

Especifica se *output_file* é armazenado no formato Unicode, independentemente do formato de *input_file*.

🕒 Observação

Para o utilitário `sqlcmd` (Go), o arquivo de saída Unicode gerado tem o BOM (marca de ordem de byte) Little-Endian UTF-16 gravado nele.

Opções de execução de consultas

-E

Grava scripts de entrada no dispositivo de saída padrão (`stdout`).

-I

Aplicável a: utilitário `sqlcmd` ODBC.

Define a opção de conexão `SET QUOTED_IDENTIFIER` como `ON`. Por padrão, ela é definida como `OFF`. Para obter mais informações, confira `SET QUOTED_IDENTIFIER` (Transact-SQL).

🕒 Observação

Para desabilitar o comportamento do identificador entre aspas, no utilitário `sqlcmd` (Go), adicione `SET QUOTED_IDENTIFIER OFF` aos seus scripts.

-q "cmdline query"

Executa uma consulta quando o `sqlcmd` é iniciado, mas não fecha o `sqlcmd` após a execução da consulta. Podem ser executadas consultas delimitadas por vários ponto e vírgula. Use aspas na consulta, conforme o exemplo a seguir.

No prompt de comando, digite:

```
Prompt de comando do Windows Copiar
sqlcmd -d Adventureworks2022 -q "SELECT FirstName, LastName FROM Person.Person WHERE LastNa
sqlcmd -d Adventureworks2022 -q "SELECT TOP 5 FirstName FROM Person.Person;SELECT TOP 5 Las
```

🚨 Importante

Não use o terminador `GO` na consulta.

Se `-b` for especificado junto com esta opção, `sqlcmd` será fechado com um erro. `-b` é descrito em outro lugar deste artigo.

-Q "cmdline query"

Executa uma consulta quando o `sqlcmd` é iniciado e fecha o `sqlcmd` imediatamente. Podem ser executadas consultas delimitadas por vários ponto e vírgula.

Use aspas na consulta, conforme o exemplo a seguir.

No prompt de comando, digite:

```
Prompt de comando do Windows Copiar
sqlcmd -d Adventureworks2022 -Q "SELECT FirstName, LastName FROM Person.Person WHERE LastNa
sqlcmd -d Adventureworks2022 -Q "SELECT TOP 5 FirstName FROM Person.Person;SELECT TOP 5 Las
```

🚨 Importante

Não use o terminador `GO` na consulta.

Se `-b` for especificado junto com esta opção, `sqlcmd` será fechado com um erro. `-b` é descrito em outro lugar deste artigo.

-t query_timeout

Especifica quanto segundos faltam para que um comando (ou instrução Transact-SQL) atinja o tempo limite. Essa opção define a variável de script do `sqlcmd` `SQLCMDSTATTIMEOUT`. Se um valor `query_timeout` não for especificado, o comando não atingirá o tempo limite. O `query_timeout` precisa ser um número entre `1` e `65534`. Se o valor fornecido não for numérico ou não estiver nesse intervalo, o `sqlcmd` gerará uma mensagem de erro.

🕒 Observação

O valor do tempo limite real poderá variar em relação ao valor `query_timeout` especificado em vários segundos.

-v var = value [var = value...]

Cria uma variável de script do `sqlcmd` que possa ser usada em um script do `sqlcmd`. Se o valor contiver espaços, mantenha-o entre aspas. Você pode especificar vários valores `<var>=<value>`". Se houver erros em algum dos valores especificados, o `sqlcmd` vai gerar uma mensagem de erro e será encerrado.

```
Prompt de comando do Windows Copiar
sqlcmd -v MyVar1=something MyVar2="some thing"
```

```
sqlcmd -v MyVar1=something -v MyVar2="some thing"
```

-X

Faz com que o `sqlcmd` ignore variáveis de script. Esse parâmetro é útil quando um script contém muitas instruções `INSERT` que podem conter cadeias de caracteres que têm o mesmo formato que variáveis regulares, como `${<variable_name>}`.

Opções de formato

-h *headers*

Especifica o número de linhas a imprimir entre os títulos da coluna. O padrão é imprimir títulos uma vez para cada conjunto de resultados de consulta. Essa opção define a variável de script do `sqlcmd` `SQLCMDHEADERS`. Use `-1` para especificar que os cabeçalhos não sejam impressos. Qualquer valor inválido faz com que o `sqlcmd` gere uma mensagem de erro e seja encerrado.

-k [1 | 2]

Remove todos os caracteres de controle, como tabulações e caracteres de nova linha, da saída. Esse parâmetro preserva a formatação de coluna quando os dados são retornados.

- `-k` remove caracteres de controle.
- `-k1` substitui cada caractere de controle por um espaço.
- `-k2` substitui caracteres de controle consecutivos por um único espaço.

-s *col_separator*

Especifica o caractere do separador de coluna. O padrão é um espaço em branco. Essa opção define a variável de script do `sqlcmd` `SQLCMDCOLSEP`. Para usar caracteres com um significado especial para o sistema operacional como, por exemplo, E comercial (&) ou ponto e vírgula (;), use-os entre aspas (""). O separador de coluna pode ser qualquer caractere de 8 bits.

-w *screen_width*

Especifica a largura de tela para saída. Essa opção define a variável de script do `sqlcmd` `SQLCMDCOLWIDTH`. A largura da coluna precisa ser um número maior que 8 e menor que 65536. Se a largura da coluna especificada não estiver nesse intervalo, o `sqlcmd` vai gerar uma mensagem de erro. A largura padrão é 80 caracteres. Quando uma linha de saída excede a largura de coluna especificada, ela inclui a próxima linha.

-W

Essa opção remove espaços à direita de uma coluna. Use essa opção juntamente com a opção `-s` ao preparar dados a serem exportados para outro aplicativo. Não é possível usar com as opções `-y` ou `-Y`.

-y *variable_length_type_display_width*

Define a variável de script `sqlcmd` `SQLCMDMAXVARTYPEWIDTH`. O padrão é 256. Ele limita o número de caracteres retornados para os tipos de dados com comprimento variável grande:

- `varchar(max)`
- `nvarchar(max)`
- `varbinary(max)`
- `xml`
- `UDTs` (tipos de dados definidos pelo usuário)
- `text`
- `ntext`
- `imagem`

UDTs podem ter comprimento fixo dependendo da implementação. Se esse tamanho de UDT de tamanho fixo for mais curto que `display_width`, o valor do UDT retornado não será afetado. Porém, se o tamanho for mais longo que `display_width`, a saída será truncada.

⊗ Cuidado

Use a opção `-y 0` com muito cuidado, porque isso poderá causar problemas de desempenho significativos tanto no servidor quanto na rede, dependendo do tamanho dos dados retornados.

🔗 -Y *fixed_length_type_display_width*

Define a variável de script `sqlcmd` `SQLCMDMAXFIXEDTYPEWIDTH`. O padrão é 0 (ilimitado). Limita o número de caracteres retornado para os tipos de dados a seguir:

• `char(n)` para `n > 1` e `n < 255`

- `cnchar(n)`, em que $1 \leq n \leq 8000$
- `nchar(n)`, em que $1 \leq n \leq 4000$
- `varchar(n)`, em que $1 \leq n \leq 8000$
- `nvarchar(n)`, em que $1 \leq n \leq 4000$
- `varbinary(n)`, em que $1 \leq n \leq 4000$
- `sql_variant`

Opções de relatório de erros

-b

Especifica que o `sqlcmd` é encerrado e retorna um valor `DOS_ERRORLEVEL` em caso de erro. O valor retornado à variável `ERRORLEVEL` será `1` quando a mensagem de erro do SQL Server tiver um nível de gravidade maior que 10. Caso contrário, o valor retornado será `0`. Se a opção `-v` tiver sido definida além de `-b`, o `sqlcmd` não relatara um erro se o nível de gravidade for inferior aos valores definidos usando `-v`. Arquivos em lote do prompt de comando podem testar o valor de `ERRORLEVEL` e tratar o erro adequadamente. `sqlcmd` não relata erros para o nível de severidade 10 (mensagens informativas).

Se o script `sqlcmd` tiver um comentário incorreto, erro de sintaxe ou se estiver sem uma variável de script, `ERRORLEVEL` retornará `1`.

-m *error_level*

Controla quais mensagens de erro são enviadas para `stdout`. Mensagens com um nível de severidade maior ou igual a esse nível são enviadas. Quando esse valor for definido como `-1`, todas as mensagens, incluindo mensagens informativas, serão enviadas. Não são permitidos espaços entre `-m` e `-1`. Por exemplo, `-m-1` é válido e `-m -1` é inválido.

Essa opção também define a variável de script do `sqlcmd` `SQLCMDERRORLEVEL`. Essa variável tem um padrão de `0`.

-V *error_severity_level*

Controla o nível de severidade usado para definir a variável `ERRORLEVEL`. Mensagens de erro com níveis de severidade maiores ou iguais a esse valor definem o `ERRORLEVEL`. Valores menores que 0 são reportados como `0`. Podem ser usados arquivos de lote e CMD para testar o valor da variável `ERRORLEVEL`.

Opções diversas

-a *packet_size*

Exige um pacote de tamanho diferente. Essa opção define a variável de script do `sqlcmd` `SQLCMDPACKETSIZE`. *packet_size* precisa ser um valor entre `512` e `32767`. O padrão é `4096`. Um tamanho de pacote maior pode melhorar o desempenho da execução de scripts que tenham muitas instruções Transact-SQL entre comandos `GO`. Pode-se solicitar um tamanho de pacote maior. Porém, se a solicitação for negada, o `sqlcmd` usará o padrão de servidor para tamanho de pacote.

-c *batch_terminator*

Especifica o terminador de lote. Por padrão, os comandos são encerrados e enviados para o SQL Server ao digitar a palavra `GO` sozinha em uma linha. Ao redefinir o terminador de lote, não use palavras-chave reservadas do Transact-SQL ou caracteres que tenham um significado especial para o sistema operacional, mesmo que elas sejam precedidas por uma barra invertida.

-L[c]

Lista os computadores servidor localmente configurados e os nomes dos computadores servidor que estão transmitindo na rede. Esse parâmetro não pode ser usado em combinação com outros parâmetros. O número máximo de computadores servidores que pode ser listado é 3000. Se a lista de servidores ficar truncada devido ao tamanho do buffer, será exibida uma mensagem de aviso.

Observação

Devido à natureza da transmissão em redes, o `sqlcmd` pode não receber a tempo uma resposta de todos os servidores. Assim, a lista de servidores retornada pode variar para cada invocação dessa opção.

Se for especificado o parâmetro opcional `c`, a saída aparecerá sem a linha de cabeçalho `Servers:`, e cada linha de servidor será listada sem espaços à esquerda. Esta apresentação é conhecida como saída limpa. A saída normal melhora o desempenho de processamento das linguagens dos scripts.

-p[1]

Imprime estatísticas de desempenho para cada conjunto de resultados. A tela a seguir é um exemplo

do formato das estatísticas de desempenho:

```
Saída Copiar
Network packet size (bytes): n
x xact[s]:
Clock Time (ms.): total      t1 avg      t2 (t3 xacts per sec.)
```

Em que:

- `x` = Número de transações que são processadas pelo SQL Server.
- `t1` = Tempo total para todas as transações.
- `t2` = Tempo médio de uma única transação.
- `t3` = Número médio de transações por segundo.

Todos os tempos estão em milissegundos.

Se o parâmetro opcional `1` for especificado, o formato de saída das estatísticas estará em formato separado por dois pontos, que poderá ser facilmente importado para uma planilha ou processado por um script.

Se o parâmetro opcional for qualquer valor diferente de `1`, será gerado um erro e o `sqlcmd` será encerrado.

-X[1]

Desabilita os comandos que podem comprometer a segurança do sistema quando o `sqlcmd` é executado em um arquivo em lotes. Os comandos desabilitados ainda são reconhecidos; o `sqlcmd` emite uma mensagem de aviso e continua. Se for especificado o parâmetro opcional `1`, o `sqlcmd` vai gerar uma mensagem de erro e será encerrado. Os comandos a seguir são desabilitados quando for usada a opção `-X`:

- `ED`
- `!!command`

Se for especificada a opção `-X`, isso impedirá que as variáveis de ambiente sejam transmitidas para o `sqlcmd`. Isso evita também que o script de inicialização especificado usando a variável de script `SQLCMDINI` seja executado. Para obter mais informações sobre as variáveis de script do `sqlcmd`, confira `sqlcmd - Usar o sqlcmd com variáveis de script`.

-?

Exibe a versão do `sqlcmd` e um resumo da sintaxe das opções do `sqlcmd`.

🕒 Observação

No macOS, execute `sqlcmd '-?'` (com aspas).

Comentários

As opções não precisam ser usadas na ordem mostrada na seção de sintaxe.

Quando são retornados vários resultados, o `sqlcmd` imprime uma linha em branco entre cada conjunto de resultados em um lote. Além disso, a mensagem `<x> rows affected` não é exibida quando não se aplica à instrução executada.

Para usar o `sqlcmd` interativamente, digite `sqlcmd` no prompt de comando com uma ou mais das opções descritas anteriormente neste artigo. Para obter mais informações, consulte [Usar o Utilitário sqlcmd](#)

🕒 Observação

As opções `-1`, `-Q`, `-Z` ou `-i` fazem com que o `sqlcmd` seja fechado após a execução.

O tamanho total da linha de comando do `sqlcmd` no ambiente de comando (por exemplo, `cmd.exe` ou `bash`), incluindo todos os argumentos e variáveis expandidas, é aquele determinado pelo sistema operacional subjacente.

Precedência de variável (baixa para alta)

1. Variáveis de ambiente do nível de sistema
2. Variáveis ambientais do nível de usuário.
3. Shell de comando (`SET X=Y`) definido no prompt de comando antes da execução do `sqlcmd`
4. `sqlcmd -v X=Y`
5. `Setvar X Y`

🕒 Observação

Para exibir as variáveis ambientais, no Painel de Controle, abra **Sistema** e selecione a guia **Avançado**.

Variáveis de script do sqlcmd

[Expandir a tabela](#)

Variável	Opção relacionada	R/W	Padrão
SQLCMDUSER	-U	R	""
SQLCMDPASSWORD	-P	--	""
SQLCMDSERVER	-S	R	"DefaultLocalInstance"
SQLCMDWORKSTATION	-H	R	"ComputerName"
SQLCMDDBNAME	-d	R	""
SQLCMDLOGINTIMEOUT	-l	R/W	"8" (segundos)
SQLCMDSTATTIMEOUT	-T	R/W	"0" = espere indefinidamente
SQLCMDHEADERS	-H	R/W	"0"
SQLCMDCOLSEP	-S	R/W	" "
SQLCMDCOLWIDTH	-w	R/W	"0"
SQLCMDPACKETSIZE	-a	R	"4096"
SQLCMDERRORLEVEL	-M	R/W	0
SQLCMDMAXVARTYPEWIDTH	-y	R/W	"256"
SQLCMDMAXFIXEDTYPEWIDTH	-y	R/W	"0" = ilimitado
SQLCMEEDITOR		R/W	"edit.com"
SQLCMDINI		R	""
SQLCMDUSEAAD	\- G	R/W	""

SQLCMDUSER, SQLCMDPASSWORD e SQLCMDSERVER são definidos quando :connect é usado.

R indica que o valor pode ser definido apenas uma vez durante a inicialização do programa.

R/W indica que o valor pode ser modificado usando o comando :setvar e que comandos seguintes serão influenciados pelo valor novo.

Comandos sqlcmd

Além de instruções Transact-SQL no sqlcmd, os comandos a seguir também estão disponíveis:

GO [count]	:List
[:]RESET	:Error
[:]ED	:Out
[:]!!	:Perftrace
[:]QUIT	:Connect
[:]EXIT	:On Error
:n	:Help
:ServerList	:XML [ON OFF]
:Setvar	:Listvar

Lembre-se do seguinte ao usar comandos sqlcmd :

- Todos os comandos do sqlcmd, exceto GO, precisam ser prefixados com dois-pontos (:).

📌 Importante

Para manter a compatibilidade com versões anteriores de scripts osql, alguns dos comandos serão reconhecidos sem os dois-pontos, o que é indicado por : .

- Os comandos sqlcmd serão reconhecidos apenas se aparecerem no início de uma linha.
- Todos os comandos sqlcmd não diferenciam maiúsculas de minúsculas.
- Cada comando deve estar em uma linha separada. Um comando não pode ser seguido por uma

Cada comando deve estar em uma linha separada. Em comandos não pode ser seguido por uma instrução Transact-SQL nem por outro comando.

- Comandos são executados imediatamente. Eles não são colocados no buffer de execução como instruções Transact-SQL.

Comandos de edição

[:]ED

Inicie o editor de textos. Esse editor pode ser usado para editar o lote Transact-SQL atual ou o último lote executado. Para editar o último lote executado, o comando `ED` deve ser digitado imediatamente depois da execução do último lote.

O editor de texto é definido pela variável de ambiente `SQLCMDEDITOR`. O editor padrão é `Edit`. Para alterar o editor, defina a variável de ambiente `SQLCMDEDITOR`. Por exemplo, para definir o editor como o Microsoft Notepad, no prompt de comando, digite:

```
SET SQLCMDEDITOR=notepad
```

[:]RESET

Desmarca o cache de instruções.

:List

Imprime o conteúdo do cache de instrução.

Variáveis

:Setvar <var> ["value"]

Define as variáveis de script do `sqlcmd`. Variáveis de script têm o seguinte formato: `$(VARIABLE)`.

Nomes de variáveis não diferenciam maiúsculas de minúsculas.

Variáveis de script podem ser definidas da seguinte forma:

- Usando-se implicitamente uma opção de linha de comando. Por exemplo, a opção `-I` define a variável `SQLCMDLOGINTIMEOUTsqlcmd`.
- Explicitamente, usando o comando `:Setvar`.
- Definindo uma variável de ambiente antes de executar o `sqlcmd`.

🕒 Observação

A opção `-X` previne que variáveis de ambiente sejam passadas para o `sqlcmd`.

Se uma variável definida com `:Setvar` e uma variável de ambiente tiverem o mesmo nome, a variável definida com `:Setvar` terá precedência.

Nomes de variáveis não devem conter caracteres de espaço em branco.

Os nomes de variáveis não podem ter a mesma forma que uma expressão variável, como `$(var)`.

Se o valor da cadeia de caracteres da variável de script tiver espaços em branco, use aspas. Se não for especificado um valor para uma variável de script, a variável de script será removida.

:Listvar

Exibe uma lista das variáveis de script definidas atualmente.

🕒 Observação

Serão exibidas somente variáveis de script definidas pelo `sqlcmd` e aquelas definidas usando o comando `:Setvar`.

Comandos de saída

:Error <filename> | STDERR | STDOUT

Redireciona toda a saída de erro para o arquivo especificado por *nome do arquivo* como `stderr` ou `stdout`. O comando `:Error` pode aparecer várias vezes em um script. Por padrão, saída de erro é enviada para `stderr`.

- *filename*

Cria e abre um arquivo que receberá a saída. Se o arquivo já existir, será truncado para zero

bytes. Se o arquivo não estiver disponível devido a permissões ou outras razões, a saída não será alternada e será enviada ao último destino especificado ou ao destino padrão.

- **STDERR**

Muda a saída de erro para o fluxo `stderr`. Se houver redirecionamento, o destino para o qual o fluxo foi redirecionado receberá a saída de erro.

- **STDOUT**

Muda a saída de erro para o fluxo `stdout`. Se houver redirecionamento, o destino para o qual o fluxo foi redirecionado receberá a saída de erro.

:Out <filename> | STDERR | STDOUT

Cria e redireciona todos os resultados da consulta para o arquivo especificado por *nome do arquivo* para `stderr` ou `stdout`. Por padrão, a saída é enviada para `stdout`. Se o arquivo já existir, será truncado para zero bytes. O comando `:Out` pode aparecer várias vezes em um script.

:Perftrace <filename> | STDERR | STDOUT

Cria e redireciona todas as informações de rastreamento de desempenho para o arquivo especificado por *nome do arquivo* para `stderr` ou `stdout`. Por padrão a saída de rastreamento de desempenho é enviada para `stdout`. Se o arquivo já existir, será truncado para zero bytes. O comando `:Perftrace` pode aparecer várias vezes em um script.

Comandos de controle de execução

:On Error [exit | ignore]

Define a ação a ser executada no caso de um erro durante a execução de script ou em lote.

Quando a opção `exit` é usada, o `sqlcmd` é encerrado com o valor de erro adequado.

Quando a opção `ignore` é usada, o `sqlcmd` ignora o erro e continua executando o lote ou script. Por padrão, é impressa uma mensagem de erro.

[:]QUIT

Faz com que o `sqlcmd` seja fechado.

[:]EXIT [(statement)]

Permite que você use o resultado de uma instrução `SELECT` como o valor retornado de `sqlcmd`. Se numérico, a primeira coluna da última linha do resultado será convertida em um inteiro de 4 bytes (longo). O MS-DOS, o Linux e o macOS transmitem o byte baixo para o processo pai ou para o nível de erro do sistema operacional. O Windows 2000 e versões posteriores transmitem todo o inteiro de 4 bytes. A sintaxe é `:EXIT(query)`.

Por exemplo:

```
text Copiar  
:EXIT(SELECT @@ROWCOUNT)
```

É possível incluir também o parâmetro `:EXIT` como parte de um arquivo em lote. Por exemplo, no prompt de comando, digite:

```
sqlcmd -Q " :EXIT(SELECT COUNT(*) FROM '%1')"
```

O utilitário `sqlcmd` envia tudo entre os parênteses () para o servidor. Se um procedimento armazenado de sistema selecionar um conjunto e retornar um valor, somente a seleção será retornada. A instrução `:EXIT()` sem nada entre os parênteses executa tudo antes dela no lote e é encerrada sem um valor retornado.

Quando uma consulta incorreta é especificada, o `sqlcmd` é encerrado sem um valor retornado.

Eis uma lista de formatos `EXIT`:

- `:EXIT`

Não executa o lote, é encerrado imediatamente e não retorna valor nenhum.

- `:EXIT()`

Executa o lote e então sai imediatamente e não retorna valor algum.

- `:EXIT(query)`

Executa o lote que inclui a consulta, e então sai depois de retornar os resultados da consulta.

Se `RAISERROR` for usado em um script do `sqlcmd` e ocorrer um estado 127, o `sqlcmd` será encerrado e retornará a ID da mensagem para o cliente. Por exemplo:

```
text Copiar
RAISERROR(50001, 10, 127)
```

Esse erro fará com que o script do `sqlcmd` seja encerrado e retorne a ID de mensagem 50001 ao cliente.

Os valores retornados `-1` a `-99` são reservados pelo SQL Server e o `sqlcmd` define os seguintes valores retornados adicionais:

[Expandir a tabela](#)

Valor retornado	Descrição
-100	Erro encontrado antes da seleção do valor de retorno.
-101	Nenhuma linha encontrada ao se selecionar o valor de retorno.
-102	Erro de conversão ao selecionar valor de retorno.

GO [count]

`GO` sinaliza tanto o término de um lote quanto a execução de qualquer instrução Transact-SQL em cache. O lote é executado várias vezes como lotes separados. Você não pode declarar uma variável mais de uma vez em um lote.

Comandos variados

`:r <filename>`

Analisa as instruções Transact-SQL e os comandos do `sqlcmd` adicionais do arquivo especificado por `filename` no cache de instruções. `filename` é lido em relação ao diretório de inicialização no qual o `sqlcmd` foi executado.

Se o arquivo contiver instruções Transact-SQL que não são seguidas por `GO`, insira `GO` na linha após `:r`.

O arquivo será lido e executado depois que for encontrado um terminador de lote. Podem ser emitidos vários comandos `:r`. O arquivo pode incluir qualquer comando `sqlcmd`, incluindo o terminador de lote `GO`.

Observação

A contagem de linha que é exibida em modo interativo será aumentada em uma para cada comando `:r` encontrado. O comando `:r` aparecerá na saída do comando de lista.

`:ServerList`

Lista os servidores configurados localmente e os nomes dos servidores que estão transmitindo na rede.

`:Connect server_name[instance_name] [-I timeout] [-U user_name [-P password]]`

Conecta-se a uma instância do SQL Server. Além disso fecha a conexão atual.

Opções de tempo limite:

[Expandir a tabela](#)

Valor	Comportamento
0	Esperar para sempre
n>0	Esperar por n segundos

A variável de script `SQLCMDSERVER` reflete a conexão ativa atual.

Se `timeout` não for especificado, o valor da variável `SQLCMDLOGINTIMEOUT` será o padrão.

Se apenas `user_name` for especificado (como uma opção ou variável de ambiente), será solicitado que o usuário insira uma senha. Os usuários não serão solicitados se as variáveis de ambiente `SQLCMDUSER` ou `SQLCMDPASSWORD` forem definidas. Se você não fornecer opções ou variáveis de ambiente, o modo de Autenticação do Windows será usado para se conectar. Por exemplo, para conectar-se a uma instância, `instance1`, do SQL Server, `myserver`, usando a segurança integrada você usaria o seguinte comando:

```
text Copiar
:connect myserver\instance1
```


Para conectar-se à instância padrão do `myserver` usando variáveis de script, você usaria as seguintes configurações:

```
text Copiar

:setvar myusername test
:setvar myservername myserver
:connect $(myservername) $(myusername)
```

[:]!! *command*

Executa comandos de sistema operacional. Para executar um comando do sistema operacional, inicie uma linha com dois pontos de exclamação (`!!`) seguida do comando do sistema operacional. Por exemplo:

```
text Copiar

:!! dir
```

🕒 Observação

O comando é executado no computador em que o `sqlcmd` está sendo executado.

:XML [ON | OFF]

Para saber mais, confira [Formato de saída XML](#) e [Formato de saída JSON](#) neste artigo.

:Help

Lista os comandos `sqlcmd`, juntamente com uma breve descrição de cada comando.

Nomes de arquivos sqlcmd

Arquivos de entrada dos `sqlcmd` podem ser especificados com a opção `-i` ou o comando `:r`.

Arquivos de saída podem ser especificados com a opção `-o` ou os comandos `:Error`, `:Out` e

`:Perftrace`. A seguir algumas diretrizes sobre como trabalhar com esses arquivos:

- `:Error`, `:Out` e `:Perftrace` precisam usar valores de *filename* separados. Se o mesmo *filename* for usado, as entradas dos comandos poderão ser misturadas.
- Se um arquivo de entrada em um servidor remoto for chamado do `sqlcmd` em um computador local e o arquivo tiver um caminho de arquivo de unidade como `:Out c:\OutputFile.txt`, o arquivo de saída será criado no computador local e não no servidor remoto.
- Dentre os caminhos de arquivo válidos estão: `C:\<filename>`, `\\<Server>\<Share>\<filename>` e `"C:\Some Folder\<file name>"`. Se houver um espaço no caminho, use aspas.
- Cada nova sessão do `sqlcmd` substitui os arquivos com nomes iguais.

Mensagens informativas

O `sqlcmd` imprime qualquer mensagem informativa enviada pelo servidor. No exemplo a seguir, depois que as instruções Transact-SQL são executadas, é impressa uma mensagem informativa.

Inicie o `sqlcmd`. No prompt de comando do `sqlcmd`, digite a consulta:

```
Console Copiar

USE AdventureWorks2022;
GO
```

Quando você pressiona `ENTER`, a seguinte mensagem informativa é impressa:

```
Saída Copiar

Changed database context to 'AdventureWorks2022'.
```

Formato de saída das consultas do Transact-SQL

O `sqlcmd` imprime, em primeiro lugar, um cabeçalho de coluna com os nomes de coluna especificados na lista de seleção. Os nomes de coluna são separados usando o caractere `SQLCMDCOLSEP`. Por padrão, esse é um espaço. Se o nome de coluna for mais curto do que a largura de coluna, a saída será preenchida com espaços até a coluna seguinte.

Essa linha é seguida por uma linha divisória formada por uma série de traços. A saída a seguir mostra um exemplo.

Inicie o `sqlcmd`. No prompt de comando do `sqlcmd`, digite a consulta:

```
Console Copiar
```

```
USE AdventureWorks2022;
SELECT TOP (2) BusinessEntityID, FirstName, LastName
FROM Person.Person;
GO
```

Quando você pressiona **ENTER**, o conjunto de resultados a seguir é retornado.

BusinessEntityID	FirstName	LastName
285	Syed	Abbas
293	Catherine	Abel

(2 row(s) affected)

Embora a coluna `BusinessEntityID` tenha apenas quatro caracteres de largura, ela foi expandida para acomodar o nome de coluna mais longo. Por padrão, a saída é finalizada com 80 caracteres. Essa largura pode ser alterada usando a opção `-w` ou definindo a variável de script `SQLCMDCOLWIDTH`.

formato de saída XML

A saída XML é o resultado de uma cláusula `FOR XML`, não formatada, em um fluxo contínuo.

Quando você esperar uma saída XML, use o seguinte comando: `:XML ON`.

Observação

O `sqlcmd` retorna mensagens de erro no formato habitual. As mensagens de erro também são produzidas no fluxo de texto XML em formato XML. Usando `:XML ON`, o `sqlcmd` não exibe mensagens informativas.

Para definir o modo XML como desativado, use o seguinte comando: `:XML OFF`.

O comando `GO` não deve ser exibido antes que o comando `:XML OFF` seja emitido, pois o comando `:XML OFF` retorna o `sqlcmd` para a saída orientada por linhas.

Dados XML (em fluxo) e dados de conjunto de linhas não podem ser misturados. Se o comando `:XML ON` for emitido antes da execução de uma instrução Transact-SQL que gera fluxos XML, a saída será distorcida. Se o comando `:XML ON` for emitido, não será possível executar instruções Transact-SQL que gerem conjuntos de linhas regulares.

Observação

O comando `:XML` não oferece suporte para a instrução `SET STATISTICS XML`.

Formato da saída JSON

Quando você espera uma saída JSON, use o seguinte comando: `:XML ON`. Caso contrário, a saída incluirá o nome da coluna e o texto JSON. Essa saída não é o JSON válido.

Para definir o modo XML como desativado, use o seguinte comando: `:XML OFF`.

Para saber mais, confira [Formato de saída XML](#) neste artigo.

Usar a autenticação do Azure AD

Exemplos de uso da autenticação do Azure AD:

```
Prompt de comando do Windows

sqlcmd -S Target_DB_or_DW.testsrv.database.windows.net -G -I 30

sqlcmd -S Target_DB_or_DW.testsrv.database.windows.net -G -U bob@contoso.com -P MyAzureADPa
```

Práticas recomendadas do sqlcmd

Use as seguintes práticas para ajudar a maximizar a segurança e a eficiência.

- Use segurança integrada.
- Use `-x[1]` em ambientes automatizados.
- Proteja arquivos de entrada e de saída usando permissões adequadas de sistema de arquivos.
- Para aumentar o desempenho, faça o máximo possível em uma sessão `sqlcmd`, em vez de usar uma série de sessões.
- Defina valores mais altos de tempo limite para execução em lote ou de consulta do que você imagina que levará para a execução em lote ou de consulta.

Use as seguintes práticas para ajudar a maximizar a exatidão:

- Use o `-v16` para registrar quaisquer [mensagens de nível de gravidade 16](#). As mensagens de gravidade 16 indicam erros gerais que podem ser corrigidos pelo usuário.
- Verifique o código de saída e a variável `DOS_ERRORLEVEL` depois que o processo for encerrado. O `sqlcmd` retorna `0` normalmente, caso contrário ele define o `ERRORLEVEL` conforme definido por `-v`. Em outras palavras, `ERRORLEVEL` não deve ser o mesmo valor que o número de erro relatado pelo SQL Server. O número do erro é um valor específico do SQL Server correspondente à função do sistema `@@ERROR`. `ERRORLEVEL` é um valor específico do `sqlcmd` para indicar porque o `sqlcmd` foi encerrado e o valor é influenciado pelo argumento da linha de comando `-b` especificado.

Usar o `-v16` em combinação com a verificação do código de saída e `DOS_ERRORLEVEL` pode ajudar a detectar erros em ambientes automatizados, especialmente portões de qualidade antes de uma versão de produção.

Conteúdo relacionado

- [Saiba mais sobre o novo utilitário go-sqlcmd no GitHub](#)
- [Início Rápido: Executar imagens de contêiner do SQL Server Linux com o Docker](#)
- [sqlcmd: iniciar o utilitário](#)
- [sqlcmd – Executar arquivos de script do Transact-SQL](#)
- [sqlcmd – usar o utilitário](#)
- [sqlcmd – Usar com variáveis de script](#)
- [sqlcmd – Conectar ao mecanismo de banco de dados](#)
- [Editar scripts SQLCMD com o Editor de Consultas](#)
- [Gerenciar etapas de trabalho](#)
- [Criar uma etapa de trabalho CmdExec](#)

[Português \(Brasil\)](#) [Suas opções de privacidade](#) [Tema](#)

[Gerenciar cookies](#) [Versões anteriores](#) [Blog](#) [Contribuir](#) [Privacidade](#) [Termos de Uso](#) [Marcas Comerciais](#) © Microsoft 2023