Architektura i klient

Instancja, bazy systemowe, pliki MDF, NDF i LDF, tryby recovery oraz narzędzia klienckie.

- SQL Server: od Sybase do Optimy w każdej księgowości
- Architektura MSSQL: instancja, baza, pliki MDF/NDF/LDF
- sqlcmd i narzędzia klienckie

SQL Server: od Sybase do Optimy w każdej księgowości

Poziom: podstawy **Dotyczy: SQL Server 2016-2022** **Czas: ~15 min**

Ta strona otwiera książkę „MSSQL": jak silnik napisany przez Sybase trafił do Microsoftu, dlaczego rozstanie obu firm wciąż widać w języku T-SQL i co czeka na kolejnych stronach.

Skróty użyte na tej stronie (8)

Skrót	Co znaczy
IA-64	Intel Itanium Architecture: 64-bitowa architektura procesorów Intela, którą obsługę dodano w SQL Server 2000
IBM	International Business Machines: firma, w której laboratorium powstał model relacyjny i pierwszy działający silnik SQL
IT	Information Technology: dział albo praca związana z komputerami i infrastrukturą firmy
MSSQL	Microsoft SQL Server: silnik bazy danych Microsoftu, na nim w Zielonej stoi księgowość Optima
NT	pierwotnie kryptonim N-Ten, od procesora, dla którego system miał powstać; popularnie dziś odczytywane jako New Technology, choć to rozwinięcie nie jest oficjalne: rdzeń dzisiejszego Windows Server
OS/2	system operacyjny firmy IBM i Microsoft z lat 80., z którego pomysłów wyrósł początkowo Windows NT
SQL	Structured Query Language: język zapytań do relacyjnych baz danych
T-SQL	Transact-SQL: dialekt SQL używany w Microsoft SQL Server, wywodzący się z języka zapytań oryginalnego silnika Sybase

Skąd się wzięło

SQL Server nie powstał w Microsoftcie. Napisała go firma Sybase, założona w 1984 roku, która zbudowała własny silnik relacyjny pod nazwą Sybase SQL Server, skierowany od początku na rynek

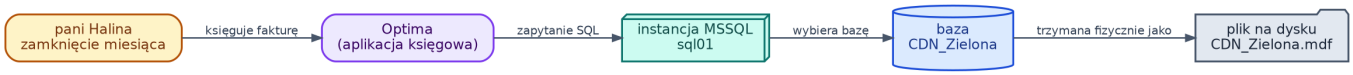
dla systemu Unix. W 1988 roku Microsoft, Sybase i firma Ashton-Tate, znana wtedy z bazy dBASE, ogłosiły wspólny projekt: wersję tego silnika dla systemu OS/2, systemu operacyjnego rozwijanego przez Microsoft razem z IBM. Rok później, w 1989 roku, ta wspólna praca trafiła na rynek jako pierwszy Microsoft SQL Server, szesnastobitowy silnik konkurujący wtedy z Oracle, IBM i Informixem.

Współpraca trwała kilka lat, ale drogi obu firm zaczęły się rozchodzić wraz z premierą Windows NT w 1993 roku: Microsoft chciał silnika dopasowanego do własnego systemu operacyjnego, Sybase rozwijał go dalej pod Unixa. W wyniku rozejścia Microsoft zatrzymał wyłączne prawa do wszystkich przyszłych wersji SQL Server na systemach Microsoftu, a Sybase przemianował swój produkt na Adaptive Server Enterprise w 1996 roku, żeby uniknąć pomyłki z produktem byłego partnera.

Kolejne lata to stopniowe odchodzenie od wspólnego kodu. SQL Server 7, wydany w 1998 roku, był dużym przepisaniem silnika w C++, z nowym rozmiarem strony danych i nowym modelem obsługi wątków. SQL Server 2000 rozwinął go dalej o obsługę architektury IA-64 i rozszerzenia języka zapytań. Ślad wspólnych korzeni z Sybase widać do dziś w samym języku zapytań: T-SQL, dialekt SQL używany w SQL Server, wywodzi się z tego samego języka, którym posługiwał się oryginalny silnik Sybase. Do wersji SQL Server 2005 stary kod Sybase został już całkowicie przepisany, a silnik stał się w pełni niezależnym produktem Microsoftu, rozwijanym dziś w edycjach od darmowego Express po Enterprise dla największych wdrożeń.

Jak to działa w jednym obrazku

Droga zapytania w SQL Server jest krótka i przewidywalna: aplikacja łączy się z instancją, instancja wybiera właściwą bazę danych, a wynik tej pracy leży fizycznie jako plik na dysku serwera.



Optima wysyła zapytanie SQL do instancji MSSQL na sql01, która wybiera bazę CDN_Zielona zapisaną fizycznie jako plik CDN_Zielona.mdf (kliknięcie otwiera pełny rozmiar)

Dlaczego Zielona

> **Z ŻYCIA WZIĘTE**
>
> W ostatni piątek zamknięcia miesiąca pani Halina dzwoni zawsze o tej samej porze: „Optima nie działa". Zwykle okazuje się, że log transakcyjny na `sql01` urósł do granicy dysku, bo nikt

> w ciągu miesiąca nie zrobił backupu logu, a baza `CDN_Zielona` pracuje w trybie `FULL`. Zespół IT nauczył się już sprawdzać wolne miejsce na dysku bazy danych na dwa dni przed każdym zamknięciem miesiąca, nie w dniu telefonu.

Optima, program księgowy kupiony razem z licencją, wymaga MSSQL i nikt w Zielonej nie negocjował tego wyboru: silnik przyszedł razem z aplikacją. Na `sql01` stoi więc baza `CDN_Zielona`, do której poza samą Optimą nikt bezpośrednio nie pisze, a jej rytm pracy wyznacza kalendarz księgowości, nie kalendarz IT. To odróżnia SQL Server od PostgreSQL na `pg01`: tu decyzję o silniku podjął dostawca oprogramowania, administrator dostaje w spadku już gotowy wybór i musi się nauczyć żyć z jego konsekwencjami.

Mapa tej książki

Książka „MSSQL" składa się z sześciu rozdziałów, budowanych na słowniku z książki „Bazy danych: fundamenty".

Rozdział	Na jakie pytanie odpowiada
SQL Server: od Sybase do Optimy w każdej księgowości	Skąd wziął się SQL Server i dlaczego Optima wymaga akurat tego silnika
Architektura i klient	Czym jest instancja, bazy systemowe, pliki MDF, NDF i LDF
Rozpoznanie i diagnostyka	Jak sprawdzić wersję, edycję, rozmiary, sesje, blokady i indeksy
Tryby pracy i log transakcyjny	Jak przełączać tryby bazy i co robić z plikiem LDF
Backup i odtwarzanie	Jak zrobić kopię zapasową, odtworzyć bazę i naprawić uprawnienia po restore
Zadania i replikacja	SQL Agent, log shipping i Change Data Capture
Procedury	Zmiana edycji silnika i operacje na bazach aplikacji

Powiązane: Baza danych, silnik, instancja i schemat · Architektura MSSQL, instancja, baza, pliki MDF/NDF/LDF · Change Data Capture (CDC): śledzenie zmian w tabelach · Zmiana edycji silnika MSSQL

Źródła: Wikipedia, „History of Microsoft SQL Server" · Microsoft SQL Server Blog, „The History of SQL Server" · Licenseware, „Microsoft SQL Server Version History" · LearnSQL, „A Brief History of MS SQL Server"

Architektura MSSQL: instancja, baza, pliki MDF/NDF/LDF

Poziom: podstawy **Dotyczy: SQL Server 2016–2022** **Czas: ~15 min**

Z czego składa się instalacja SQL Server — instancja, usługi, bazy systemowe, pliki na dysku — i czym różnią się od siebie tryby recovery oraz edycje silnika.

Skróty użyte na tej stronie (5)

Skrót	Co znaczy
RPO	Recovery Point Objective: ile danych wolno stracić przy awarii
SQL	Structured Query Language: język zapytań do relacyjnych baz danych
SSMS	SQL Server Management Studio: graficzna konsola MSSQL
TCP/IP	zestaw protokołów internetu: IP przenosi pakiety między adresami, TCP pilnuje, żeby dane doszły w całości i w kolejności
UDP	User Datagram Protocol: szybki protokół sieciowy bez potwierdzeń

Fiszka

TYLKO ODCZYT

Sprawdzenie, jakie instancje MSSQL są zainstalowane na serwerze

```
Get-Service -Name "MSSQL*"
```

Sprawdzamy, ile instancji silnika działa na danym serwerze Windows i pod jakimi nazwami usług. Nazwa usługi instancji domyślnej to zawsze `MSSQLSERVER`, a instancji nazwanej `MSSQL$NazwaInstancji`.

Przykład:

```
Status   Name               DisplayName
------   ----               -----------
Running  MSSQLSERVER        SQL Server (MSSQLSERVER)
Running  MSSQL$SQLBI        SQL Server (SQLBI)
Running  SQLSERVERAGENT     SQL Server Agent (MSSQLSERVER)
Running  SQLBrowser         SQL Server Browser
```

MSSQL = Microsoft SQL Server: silnik bazy danych Microsoftu, na nim w Zielonej stoi księgowość Optima

TYLKO ODCZYT

Sprawdzenie nazwy instancji i serwera z poziomu T-SQL

```
SELECT @@SERVERNAME AS NazwaSerwera, SERVERPROPERTY('InstanceName') AS NazwaInstancji;
```

Sprawdzamy, do jakiej instancji jesteśmy akurat podłączeni — przydatne, kiedy na jednym serwerze fizycznym siedzi kilka instancji naraz i łatwo pomylić się w SSMS.

Przykład:

```
NazwaSerwera        NazwaInstancji
------------------- --------------
SQL01\SQLBI       SQLBI


(1 rows affected)
```

T-SQL = Transact-SQL: dialekt SQL używany w Microsoft SQL Server, wywodzący się z języka zapytań oryginalnego silnika Sybase · **SSMS** = SQL Server Management Studio: graficzna konsola MSSQL

Z życia wzięte

Z ŻYCIA WZIĘTE

W poniedziałek o 8:15 Tomek próbuje połączyć się z bazą CDN_Zielona, wpisując w SSMS samą nazwę sql01, tak jak w starej instrukcji od poprzedniego administratora. Połączenie

kończy się błędem, chociaż serwer odpowiada na ping, a księgowość normalnie loguje się do Optimy. Dopiero sprawdzenie usług Windows pokazuje, że silnik dla CDN_Zielona działa jako instancja nazwana sql01\SQLBI, nie jako instancja domyślna. Po dopisaniu nazwy instancji do connection stringa Tomek łączy się bez problemu.

Po co to istnieje

Pojęcie: instancja. Osobna, samodzielna kopia silnika SQL Server działająca na tym samym serwerze fizycznym albo wirtualnym co inne instancje, ale ze swoim własnym zestawem baz systemowych, swoją konfiguracją pamięci, swoim portem i swoim kontem usługi. Instancje nie widzą nawzajem swoich baz — połączenie do jednej z nich nie daje dostępu do baz drugiej.

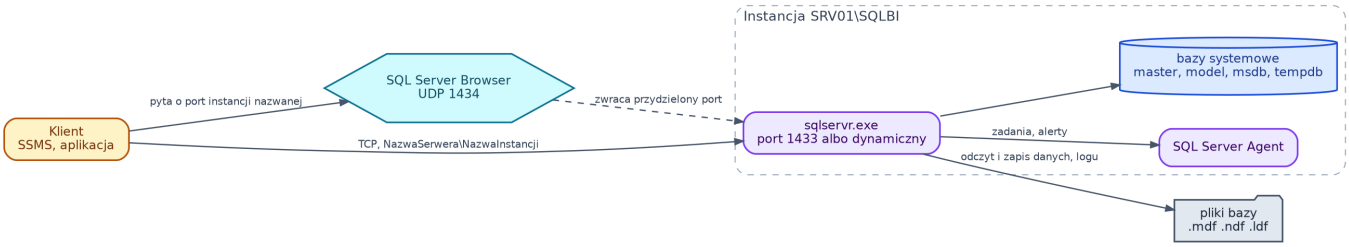
Pojęcie: instancja domyślna i instancja nazwana. Na serwerze może działać dokładnie jedna instancja domyślna, adresowana samą nazwą serwera (`SQL01`), oraz dowolna liczba instancji nazwanych, adresowanych jako `NazwaSerwera\NazwaInstancji` (`SQL01\SQLBI`). Instancja domyślna nasłuchuje zawsze na porcie 1433, instancje nazwane — na porcie przydzielonym dynamicznie, chyba że ustawimy port na stałe w Configuration Managerze.

Rozróżnienie instancja / baza danych / silnik jest tym, co najczęściej myli się na starcie: silnik to program `sqlservr.exe`, instancja to jedno uruchomienie tego programu ze swoim kontekstem, a baza danych to pojedynczy zbiór danych żyjący wewnątrz instancji. Jedna instancja trzyma zwykle wiele baz — systemowe plus te, które są przedmiotem naszej pracy.

Jak to działa

Rozwiń: jak to działa pod spodem

Pojęcie: usługi SQL Server. Instalacja silnika stawia na Windows kilka usług, z których każda odpowiada za inny fragment funkcjonalności.



Klient pyta SQL Server Browser o port instancji nazwanej, po czym łączy się z sqlservr.exe, który korzysta z baz systemowych i zapisuje dane do plików mdf, ndf i ldf (kliknięcie otwiera pełny rozmiar)

Usługa	Nazwa procesu / usługi	Za co odpowiada
SQL Server (silnik)	`sqlservr.exe`, `MSSQLSERVER` lub `MSSQL$NazwaInstancji`	wykonywanie zapytań, przechowywanie danych, logi transakcyjne
SQL Server Agent	`SQLSERVERAGENT` lub `SQLAGENT$NazwaInstancji`	zadania harmonogramowane, alerty, powiadomienia
SQL Server Browser	`SQLBrowser`	odpowiada klientom, na jakim porcie nasłuchuje dana instancja nazwana
SQL Server VSS Writer	`SQLWriter`	integracja z backupem na poziomie systemu plików (VSS)

Pojęcie: port TCP silnika. Instancja domyślna nasłuchuje standardowo na porcie 1433/TCP. Instancje nazwane dostają port dynamiczny przydzielany przy każdym starcie usługi, chyba że administrator wymusi port stały w Configuration Managerze, w zakładce protokołu TCP/IP. Klient, który zna tylko nazwę instancji, a nie zna portu, pyta o niego usługę SQL Server Browser działającą na porcie UDP 1434 — to dlatego wyłączenie tej usługi łamie połączenia po nazwie instancji, mimo że sam silnik działa poprawnie.

Pojęcie: bazy systemowe. Każda instancja, zaraz po instalacji, ma cztery bazy systemowe, które trzymają metadane samej instancji i nie są przedmiotem naszej pracy z danymi biznesowymi.

Baza	Co trzyma
`master`	konfigurację instancji, listę wszystkich baz danych, loginy na poziomie serwera, ustawienia konfiguracyjne
`model`	szablon, z którego kopiowana jest każda nowo tworzona baza — zmiana w `model` wpływa na wszystkie kolejne `CREATE DATABASE`

Baza	Co trzyma
`msdb`	historię i definicje zadań SQL Server Agent, historię backupów i restore'ów, plany konserwacyjne, alerty
`tempdb`	tabele tymczasowe, sortowania na dysku, wersje wierszy pod izolację `READ COMMITTED SNAPSHOT` — jedyna baza systemowa tworzona od nowa przy każdym starcie usługi

> **Pojęcie: plik danych i plik logu.** Fizycznie baza danych to zestaw plików na dysku, nigdy jeden monolityczny plik.

Rozszerzenie	Nazwa	Rola
`.mdf`	primary data file	pierwszy, obowiązkowy plik danych — trzyma też katalog systemowy bazy
`.ndf`	secondary data file	dodatkowe pliki danych, opcjonalne, używane do rozłożenia danych na kilka dysków albo grup plików
`.ldf`	log file	plik logu transakcyjnego — sekwencyjny zapis każdej zmiany, zanim trafi ona do plików danych

> **Pojęcie: grupa plików (filegroup).** Logiczny kontener grupujący jeden lub więcej plików danych (`.mdf` / `.ndf`), do którego przypisujemy tabele i indeksy. Grupa `PRIMARY` istnieje zawsze i zawiera plik `.mdf`. Dodatkowe grupy pozwalają rozłożyć duże tabele na osobne dyski albo trzymać archiwalne partycje na wolniejszym storage'u, bez zmiany ani jednego zapytania — to `CREATE TABLE ... ON NazwaGrupyPlikow` decyduje, gdzie fizycznie wyląduje tabela.

> **Pojęcie: autogrowth.** Automatyczne powiększanie pliku danych albo logu, kiedy zapełni się do końca, o wartość skonfigurowaną jako stały rozmiar (np. 256 MB) albo procent bieżącego rozmiaru. Domyślne wartości fabryczne (1 MB dla danych, 10% dla logu) są zbyt małe dla realnej produkcji — częste, drobne powiększenia fragmentują plik na dysku i chwilowo blokują zapis, dlatego ustawiamy autogrowth ręcznie na sensowną stałą wartość zaraz po utworzeniu bazy.

Tryby recovery

> **Pojęcie: tryb recovery (recovery model).** Ustawienie bazy danych decydujące o tym, jak długo silnik trzyma zapisy w logu transakcyjnym i jakie opcje odtwarzania są w ogóle dostępne. To jedna z pierwszych rzeczy, którą sprawdzamy na nieznanej bazie — opisane szerzej na stronie Log transakcyjny: WAL, redo, transaction log.

Tryb	Co dzieje się z logiem	Backup logu	Odtwarzanie do punktu w czasie
`SIMPLE`	log jest obcinany automatycznie po każdym checkpoincie, nie rośnie w nieskończoność	niedostępny — nie ma czego backupować	niedostępne
`FULL`	log rośnie, dopóki nie zrobimy jego backupu — dopiero backup logu zwalnia miejsce	wymagany do zachowania ciągłości odtwarzania	dostępne, z dokładnością do transakcji
`BULK_LOGGED`	operacje masowe (`BULK INSERT`, przebudowa indeksu, `SELECT INTO`) logowane są minimalnie, reszta jak w `FULL`	dostępny, ale backup logu obejmujący operację masową nie pozwala na PITR w jej trakcie	ograniczone — nie działa dla przedziału z operacją masową

Wybór trybu jest kompromisem między rozmiarem logu a możliwością odtworzenia bazy na sekundę przed awarią. Baza produkcyjna, dla której liczy się RPO bliski zeru, pracuje w `FULL` i ma regularny backup logów opisany na stronie Backup pełny, różnicowy, przyrostowy, logi. Baza testowa albo hurtownia zasilana raz dziennie, dla której strata dnia danych jest akceptowalna, może bezpiecznie pracować w `SIMPLE`.

Edycje silnika

> **Pojęcie: edycja (edition).** Wariant licencyjny silnika SQL Server, decydujący o limitach zasobów i dostępności niektórych funkcji — sprawdzenie zainstalowanej edycji opisujemy na stronie Rozpoznanie instancji i baz.

Edycja	Limit pamięci silnika	Limit rdzeni	Typowe zastosowanie
Express	1410 MB bufora	4 rdzenie	aplikacje wbudowane, małe wdrożenia, brak kosztu licencji
Standard	128 GB bufora	24 rdzenie	typowa produkcja średniej wielkości

Edycja	Limit pamięci silnika	Limit rdzeni	Typowe zastosowanie
Enterprise	brak limitu	brak limitu	duże instalacje, `ONLINE` rebuild indeksów, partycjonowanie tabel, Always On z wieloma repликami
Developer	jak Enterprise	jak Enterprise	funkcjonalnie pełna, ale licencyjnie wyłącznie do środowisk deweloperskich i testowych, nie do produkcji

Różnica między Standard a Enterprise dotyczy administratora najbardziej przy operacjach na indeksach — `ALTER INDEX ... REBUILD WITH (ONLINE = ON)` wymaga Enterprise, temat rozwinięty na stronie Indeksy: status, fragmentacja, przebudowa, oraz przy skalowaniu, gdzie limit rdzeni i pamięci w Standard bywa pierwszym wąskim gardłem, jaki napotyka rosnąca baza.

Częste błędy

Mylenie instancji z serwerem. Adres `SQL01` bez nazwy instancji łączy zawsze z instancją domyślną — jeśli na serwerze stoi wyłącznie instancja nazwana `SQL01\SQLBI`, samo `SQL01` zwróci błąd połączenia, mimo że serwer żyje i odpowiada na ping.

Traktowanie `tempdb` jak zwykłej bazy do trzymania danych roboczych na stałe. `tempdb` jest tworzona od zera przy każdym restarcie usługi — wszystko, co w niej zapisano, znika po restarcie silnika, łącznie z tabelami tymczasowymi utworzonymi ręcznie jako „stałe" miejsce robocze.

Zostawienie domyślnego autogrowth na produkcji. Plik danych rosnący o 1 MB przy bazie zapisującej gigabajty dziennie powiększa się tysiące razy, każde powiększenie to pauza w zapisie i fragmentacja pliku na poziomie systemu plików.

Powiązane: Rozpoznanie instancji i baz · Log transakcyjny: WAL, redo, transaction log · Rozmiar bazy: pliki, tablespace'y, autoextend · Indeksy: status, fragmentacja, przebudowa

Źródła: dokumentacja Microsoft Learn, sekcja „SQL Server instances" i „Database Files and Filegroups"

sqlcmd i narzędzia klienckie

Poziom: podstawy **Dotyczy: SQL Server 2016–2022, sqlcmd Utility** **Czas: ~15 min**

Łączenie się z instancją z wiersza poleceń, kiedy nie mamy pod ręką SSMS — parametry `sqlcmd`, separator wsadu `GO` i skąd biorą się okna administracyjne.

Skróty użyte na tej stronie (6)

Skrót	Co znaczy
SQL	Structured Query Language: język zapytań do relacyjnych baz danych
SSH	Secure Shell: szyfrowane zdalne logowanie do serwera
SSMS	SQL Server Management Studio: graficzna konsola MSSQL
T-SQL	Transact-SQL: dialekt SQL używany w Microsoft SQL Server, wywodzący się z języka zapytań oryginalnego silnika Sybase
TCP/IP	zestaw protokołów internetu: IP przenosi pakiety między adresami, TCP pilnuje, żeby dane doszły w całości i w kolejności
WMI	Windows Management Instrumentation: interfejs do odczytu stanu Windowsa

Fiszka

TYLKO ODCZYT

Połączenie uwierzytelnianiem Windows

```
sqlcmd -S NazwaSerwera\NazwaInstancji -E
```

Łączymy się do wskazanej instancji, używając konta, na którym akurat pracujemy w systemie Windows — flaga `-E` włącza uwierzytelnianie zintegrowane. Dla instancji domyślnej pomijamy `\NazwaInstancji` i podajemy samą nazwę serwera.

sqlcmd = SQL command: klient bazy MSSQL

TYLKO ODCZYT

Połączenie uwierzytelnianiem SQL Server

```
sqlcmd -S NazwaSerwera -U NazwaLoginu -P HasloLoginu
```

Łączymy się loginem i hasłem zdefiniowanym po stronie silnika, niezależnie od konta systemu Windows, na którym siedzimy — potrzebne, kiedy klient nie jest w tej samej domenie co serwer albo login jest kontem aplikacyjnym, a nie kontem osoby.

> **Pojęcie: tryb uwierzytelniania.** Instancja SQL Server działa w jednym z dwóch trybów: `Windows Authentication mode`, gdzie akceptowane są wyłącznie konta domenowe albo lokalne systemu Windows, albo `Mixed Mode`, gdzie dodatkowo istnieją własne loginy SQL Server z hasłem trzymanym po stronie silnika. Tryb ustawiamy we właściwościach serwera w SSMS albo kluczem rejestru `LoginMode`, a zmiana wymaga restartu usługi silnika.

SQL = Structured Query Language: język zapytań do relacyjnych baz danych · **SSMS** = SQL Server Management Studio: graficzna konsola MSSQL · **sqlcmd** = SQL command: klient bazy MSSQL

TYLKO ODCZYT

Wskazanie bazy domyślnej dla sesji

```
sqlcmd -S NazwaSerwera -E -d NazwaBazy
```

Ustawiamy, na której bazie ląduje sesja zaraz po połączeniu, zamiast startować w `master` i wykonywać ręcznie `USE NazwaBazy`.

sqlcmd = SQL command: klient bazy MSSQL

TYLKO ODCZYT

Wykonanie pojedynczego zapytania bez wchodzenia w tryb interaktywny

```
sqlcmd -S NazwaSerwera -E -d NazwaBazy -Q "SELECT @@VERSION"
```

Odpalamy jedno zapytanie i kończymy sesję zaraz po zwróceniu wyniku — flaga `-Q` (duże Q) zamyka połączenie po wykonaniu, `-q` (małe q) wykonuje zapytanie i zostawia sesję otwartą w trybie interaktywnym.

Przykład:

```
Microsoft SQL Server 2019 (RTM-CU18) (KB5017593) - 15.0.4223.1 (X64)
        Sep  8 2022 12:34:56
        Copyright (C) 2019 Microsoft Corporation
        Standard Edition (64-bit) on Windows Server 2019 Standard 10.0 <X64> (Build
17763: )

(1 rows affected)
```

sqlcmd = SQL command: klient bazy MSSQL

TYLKO ODCZYT

Wykonanie skryptu z pliku

```
sqlcmd -S NazwaSerwera -E -d NazwaBazy -i SciezkaDoPliku.sql -o SciezkaDoPliku.log
```

Uruchamiamy gotowy skrypt `.sql` w trybie wsadowym, przekierowując cały wynik do pliku logu zamiast na ekran — tak wygląda wywołanie `sqlcmd` z poziomu harmonogramu Windows albo z zewnętrznego skryptu wdrożeniowego.

sqlcmd = SQL command: klient bazy MSSQL

TYLKO ODCZYT

Ustawienie zmiennej sqlcmd i użycie jej w skrypcie

```
sqlcmd -S NazwaSerwera -E -v BazaDocelowa="NazwaBazy" -i SciezkaDoPliku.sql
```

Przekazujemy do skryptu wartość z poziomu wiersza poleceń, bez edytowania samego pliku `.sql` — wewnątrz skryptu odwołujemy się do niej jako `$(BazaDocelowa)`. Przydatne, gdy ten sam skrypt wdrożeniowy ma trafić na kilka różnych baz w kolejnych środowiskach.

sqlcmd = SQL command: klient bazy MSSQL

TYLKO ODCZYT

Wszystkie parametry połączenia

Flaga	Co robi

`-S`	serwer i opcjonalnie instancja (`NazwaSerwera\NazwaInstancji`)
`-U` / `-P`	login i hasło uwierzytelniania SQL Server
`-E`	uwierzytelnianie zintegrowane Windows
`-d`	baza domyślna dla sesji
`-Q`	wykonanie zapytania i zamknięcie połączenia
`-q`	wykonanie zapytania z pozostawieniem sesji otwartej
`-i`	plik wejściowy ze skryptem `.sql`
`-o`	plik wyjściowy na wynik
`-v`	zmienna sqlcmd podstawiana w skrypcie jako `$(NazwaZmiennej)`

SQL = Structured Query Language: język zapytań do relacyjnych baz danych

Z życia wzięte

Z ŻYCIA WZIĘTE

W piątek wieczorem pada zdalny pulpit do sql01 i Tomek zostaje tylko z dostępem SSH do serwera terminalowego, bez SSMS pod ręką. Musi sprawdzić, czy nocny job odtworzył testową kopię CDN_Zielona_TEST, więc łączy się poleceniem sqlcmd z uwierzytelnianiem zintegrowanym prosto z konsoli. Zapomina o GO po USE NazwaBazy i pierwsze zapytanie leci jeszcze w kontekście master, co przez chwilę wygląda jak pusty wynik. Po dodaniu separatora wsadu zapytanie zwraca właściwą odpowiedź z CDN_Zielona_TEST.

Po co to istnieje

Pojęcie: SSMS. SQL Server Management Studio — graficzne narzędzie do pracy z instancją: edytor zapytań, przeglądarka obiektów, kreatory backupu i restore'u, podgląd planów wykonania. Większość codziennej pracy administracyjnej dzieje się właśnie w SSMS, ale nie zawsze mamy je pod ręką — na serwerze bez pulpitu graficznego, wewnątrz skryptu automatyzującego albo w zadaniu harmonogramowanym jedynym dostępnym narzędziem jest `sqlcmd`, klient wiersza poleceń instalowany razem z silnikiem albo osobno jako `sqlcmd Utility`.

> **Pojęcie: SQL Server Configuration Manager.** Osobna przystawka `mmc` służąca do zarządzania usługami silnika, protokołami sieciowymi (TCP/IP, Named Pipes) i portami nasłuchu poszczególnych instancji. To w niej ustawiamy port stały dla instancji nazwanej albo wymuszamy szyfrowanie połączeń — zmiany tutaj wymagają zwykle restartu usługi silnika, żeby zaczęły obowiązywać.

Jak to działa

> **Pojęcie: wsad (batch) i separator `GO`.** `sqlcmd`, tak jak SSMS, wysyła polecenia do silnika w porcjach zwanych wsadami. Słowo `GO`, wpisane w osobnej linii, nie jest częścią składni T-SQL — to instrukcja dla samego klienta, żeby wysłał wszystko, co zebrał od poprzedniego `GO`, jako jeden wsad. Dlatego niektóre instrukcje (`CREATE PROCEDURE`, zmiana bazy przez `USE`) muszą stać jako pierwsze w swoim wsadzie — silnik kompiluje wsad jako całość, zanim zacznie go wykonywać.

```
USE NazwaBazy;
GO

SELECT COUNT(*) FROM NazwaTabeli;
GO
```

> **Pojęcie: `SET NOCOUNT`.** Domyślnie po każdej instrukcji modyfikującej dane silnik odsyła komunikat w rodzaju `(12 rows affected)`. `SET NOCOUNT ON` wyłącza te komunikaty dla bieżącej sesji — w procedurach składowanych wywoływanych masowo z aplikacji ogranicza to zbędny ruch sieciowy i przyspiesza wykonanie, bo aplikacja i tak zwykle ignoruje tę informację.

```
SET NOCOUNT ON;
UPDATE NazwaTabeli SET Kolumna = 1 WHERE Warunek = 1;
```

Częste błędy

Próba wykonania `USE` i zapytania w tym samym wsadzie razem z tworzeniem obiektu. `CREATE PROCEDURE` musi być pierwszą instrukcją wsadu — próba połączenia jej z `USE` w jednym bloku bez `GO` pomiędzy nimi kończy się błędem `Msg 111, Level 15, State 1, 'CREATE PROCEDURE' must be the first statement in a query batch.`

Pomylenie `-Q` z `-q`. Wielka litera zamyka sesję od razu po wykonaniu — skrypt, który liczy na to, że po `-q` będzie mógł wysłać kolejne polecenia interaktywnie, przy `-Q` dostanie zamknięte połączenie.

Naprawa uszkodzonej przystawki Configuration Manager. Po aktualizacji systemu albo instalacji kolejnej instancji przystawka Configuration Manager czasem przestaje się uruchamiać, zgłaszając błąd rejestracji dostawcy WMI. Naprawiamy to rejestrując ponownie komponent `mofcomp` z katalogu udostępnionych komponentów silnika, uruchomionego z wiersza poleceń jako administrator:

```
cd "C:\Program Files (x86)\Microsoft SQL Server\140\Shared"
mofcomp sqlmgmproviderxpsp2up.mof
```

> (gdzie `140` to numer wersji silnika w ścieżce — SQL Server 2017; dla innych wersji katalog nosi inny numer, np. `130` dla 2016 albo `150` dla 2019)

Przykład:

```
Microsoft (R) MOF Compiler Version 10.0.17763.4728
Copyright (c) Microsoft Corp. 1997-2006. All rights reserved.
Parsing MOF file: sqlmgmproviderxpsp2up.mof
MOF file has been successfully parsed
Storing data in the repository...
Done!
```

Po tym poleceniu przystawka Configuration Manager uruchamia się poprawnie.

Powiązane: Rozpoznanie instancji i baz · Architektura MSSQL: instancja, baza, pliki MDF/NDF/LDF

Źródła: dokumentacja Microsoft Learn, sekcja „sqlcmd Utility" i „SQL Server Configuration Manager"