hdbsql: klient wiersza poleceń

Poziom: podstawy **Dotyczy: SAP HANA 2.0 SPS ≥ 04** **Czas: ~20 min**

`hdbsql` odpowiada w świecie HANA temu, czym `psql` jest dla PostgreSQL albo `mysql` dla MySQL — z jedną istotną różnicą: hasła do niego w praktyce produkcyjnej w ogóle nie wpisujemy wprost.

Skróty użyte na tej stronie (5)

Skrót	Co znaczy
CLI	Command Line Interface: praca przez wpisywanie komend w terminalu
HANA	SAP HANA: baza danych w pamięci firmy SAP, używana do dużych systemów księgowych i magazynowych
MDC	Multitenant Database Container: tryb SAP HANA, w którym jedna instalacja obsługuje bazę systemową i wiele baz tenant naraz
SAP	Systeme, Anwendungen und Produkte in der Datenverarbeitung: niemiecki producent oprogramowania klasy ERP, właściciel silnika HANA i przejętego wcześniej Sybase ASE
SQL	Structured Query Language: język zapytań do relacyjnych baz danych

Fiszka

TYLKO ODCZYT

Sprawdzenie listy baz danych instalacji MDC: SYS.M_DATABASES

```
SELECT DATABASE_NAME, ACTIVE_STATUS, HOST FROM SYS.M_DATABASES;
```

Sprawdzamy, jakie bazy (System DB i tenanty) istnieją w tej instalacji i czy są aktywne — pierwsze pytanie po podłączeniu do nieznanej instalacji w trybie MDC opisanym na stronie Architektura SAP HANA: instancja, numer, nazwa SID, procesy.

Przykład:

```
DATABASE_NAME  ACTIVE_STATUS  HOST
SYSTEMDB       YES            hana01
HDB            YES            hana01
```

MDC = Multitenant Database Container: tryb SAP HANA, w którym jedna instalacja obsługuje bazę systemową i wiele baz tenant naraz · **SAP** = Systeme, Anwendungen und Produkte in der Datenverarbeitung: niemiecki producent oprogramowania klasy ERP, właściciel silnika HANA i przejętego wcześniej Sybase ASE · **HANA** = SAP HANA: baza danych w pamięci firmy SAP, używana do dużych systemów księgowych i magazynowych · **SID** = w Oracle: System Identifier, nazwa instancji; w Windows: Security Identifier, numer konta

TYLKO ODCZYT

Sprawdzenie stanu usług instancji: M_SERVICES

```
SELECT HOST, PORT, SERVICE_NAME, PROCESS_ID, ACTIVE_STATUS FROM M_SERVICES;
```

Sprawdzamy z poziomu SQL to samo, co `sapcontrol -function GetProcessList` pokazuje z poziomu systemowego — listę usług HANA wraz z portami i statusem, wygodne, gdy jesteśmy już podłączeni klientem SQL i nie chcemy przełączać się do powłoki systemowej.

Przykład:

```
HOST       PORT   SERVICE_NAME       PROCESS_ID  ACTIVE_STATUS
hana01   30001  nameserver         4102        YES
hana01   30003  indexserver        4318        YES
hana01   30007  xsengine           4402        YES
hana01   30010  compileserver      4210        YES
```

SQL = Structured Query Language: język zapytań do relacyjnych baz danych · **HANA** = SAP HANA: baza danych w pamięci firmy SAP, używana do dużych systemów księgowych i magazynowych

TYLKO ODCZYT

Sprawdzenie zajętości dysków: M_DISKS

```
SELECT PATH, USED_SIZE, TOTAL_SIZE, USAGE_TYPE FROM M_DISKS;
```

Sprawdzamy wykorzystanie wolumenów danych i logu w bajtach — pierwsze miejsce, gdzie szukamy przyczyny alertu o zapełniającym się `/hana/log`, opisanego szerzej na stronie

Diagnostyka i miejsce na dysku.

Przykład:

```
PATH             USED_SIZE      TOTAL_SIZE      USAGE_TYPE
/hana/data/HDB   214748364800   536870912000    DATA
/hana/log/HDB     42949672960    64424509440    LOG
```

TYLKO ODCZYT

Sprawdzenie aktywnych połączeń: M_CONNECTIONS

```
SELECT CONNECTION_ID, HOST, PORT, USER_NAME, CONNECTION_STATUS, CLIENT_HOST
FROM M_CONNECTIONS
WHERE CONNECTION_STATUS = 'RUNNING';
```

Sprawdzamy, kto jest aktualnie podłączony do bazy i które połączenie faktycznie wykonuje zapytanie w danej chwili, w odróżnieniu od połączeń bezczynnych.

Przykład:

```
CONNECTION_ID  HOST       PORT   USER_NAME     CONNECTION_STATUS  CLIENT_HOST
482            hana01   30003  NAZWA_USERA   RUNNING            app01
```

TYLKO ODCZYT

Wylistowanie tabel w bieżącej bazie: TABLES

```
SELECT SCHEMA_NAME, TABLE_NAME, RECORD_COUNT FROM M_TABLES WHERE SCHEMA_NAME =
'NazwaSchematu';
```

Sprawdzamy tabele danego schematu wraz z przybliżoną liczbą wierszy, bez sięgania po pełny `SELECT COUNT(*)` na każdej z osobna.

Przykład:

```
SCHEMA_NAME     TABLE_NAME    RECORD_COUNT
NAZWA_SCHEMATU  ZAMOWIENIA    48213
NAZWA_SCHEMATU  KLIENCI       9284
```

Z życia wzięte

Z ŻYCIA WZIĘTE

Ola chce podłączyć zewnętrzny skrypt monitorujący do hana01, ale nie chce trzymać hasła jawnym tekstem w cronie. Zapisuje dane logowania kluczem MONITORING w hdbuserstore i skrypt łączy się poleceniem hdbsql -U MONITORING bez podawania czegokolwiek więcej. Miesiąc później podczas przeglądu bezpieczeństwa pan Wojtek sprawdza listę procesów na serwerze i nie znajduje ani jednego hasła widocznego w ps aux, co akurat go zadowala.

Parametry połączenia

```
hdbsql -i NumerInstancji -n NazwaSerwera -u NazwaUsera -p HasloUzytkownika -d NazwaBazy
```

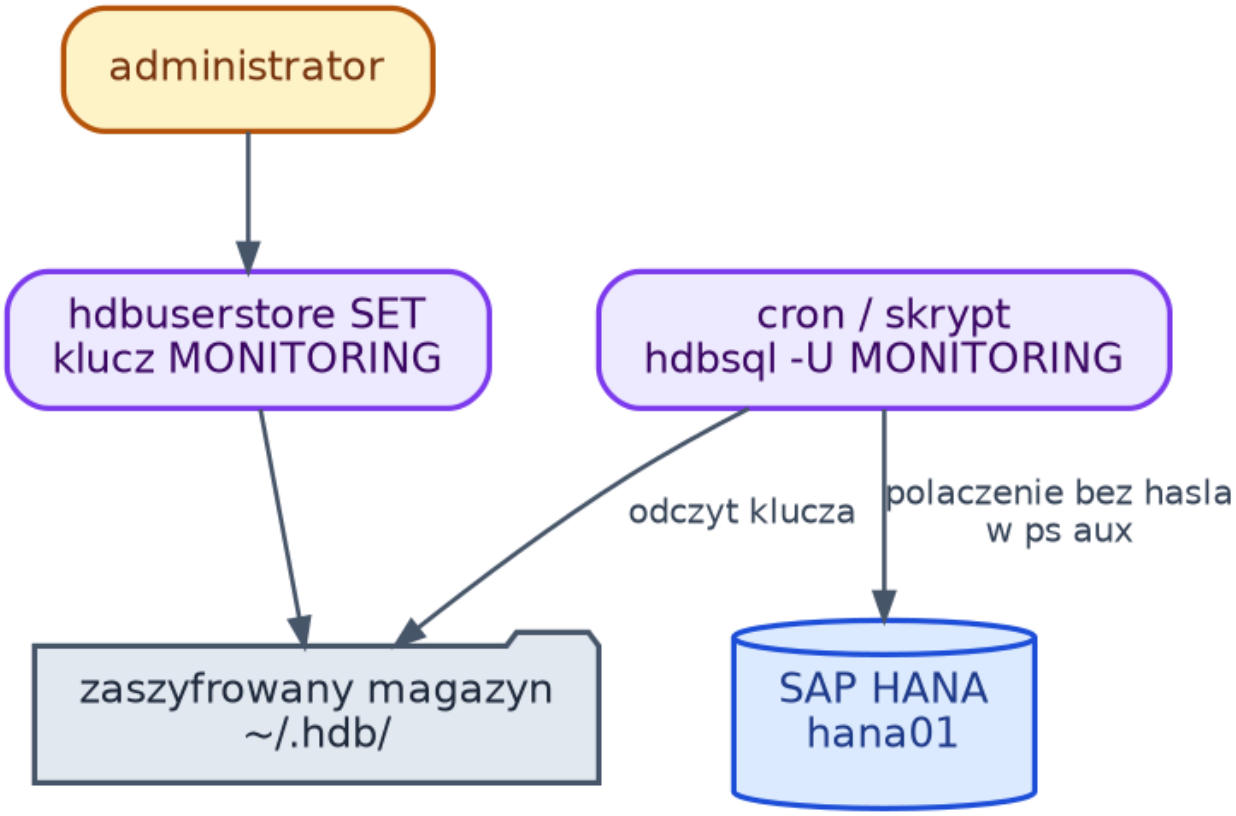
Flaga `-i` wskazuje numer instancji (nie numer portu — port HANA sam wylicza się z numeru instancji według wzoru opisanego na stronie Architektura SAP HANA: instancja, numer, nazwa SID, procesy), `-n` wskazuje host, `-u` i `-p` dane logowania, `-d` nazwę bazy docelowej — `SYSTEMDB` dla bazy systemowej albo nazwę konkretnego tenanta w instalacji MDC.

Przykład połączenia do tenanta na zdalnym hoście:

```
hdbsql -i 00 -n hana01 -u NazwaUsera -p HasloUzytkownika -d NazwaTenanta
```

hdbuserstore — dlaczego nie podajemy hasła w wierszu poleceń

Pojęcie: hdbuserstore. Narzędzie CLI zapisujące dane logowania (host, port, użytkownik, hasło) w zaszyfrowanym magazynie w katalogu domowym użytkownika systemowego, pod nazwaną kluczem referencją — odpowiednik `mysql_config_editor` z MySQL albo `.pgpass` z PostgreSQL, tylko bardziej rygorystycznie egzekwowany w środowiskach SAP, gdzie hasło w wierszu poleceń widoczne w `ps aux` jest traktowane jako naruszenie polityki bezpieczeństwa, nie tylko niedogodność.



Zapisanie klucza w hdbuserstore i połączenie skryptu z HANA bez hasła w wierszu poleceń (kliknięcie otwiera pełny rozmiar)

Zapisanie klucza:

```
hdbuserstore SET NazwaKlucza NazwaSerwera:NumerPortu@NazwaBazy NazwaUsera HasloUzytkownika
```

Przykład:

```
hdbuserstore SET MONITORING hana01:30013@SYSTEMDB nazwa_usera HasloUzytkownika
```

Wylistowanie zapisanych kluczy — bez ujawniania samych haseł:

```
hdbuserstore LIST
```

Przykład:

```
DATA FILE       : /home/nazwa_usera/.hdb/hana01/SSFS_HDB.DAT
KEY FILE        : /home/nazwa_usera/.hdb/hana01/SSFS_HDB.KEY

KEY MONITORING
```

```
  ENV : hana01:30013
  USER: NAZWA_USERA
  DATABASE: SYSTEMDB
```

Połączenie z użyciem klucza, bez podawania czegokolwiek poza samą nazwą klucza:

```
hdbsql -U NazwaKlucza
```

Ten sam mechanizm klucza wykorzystują zadania `cron` i skrypty backupu, o których mowa na stronie Diagnostyka i miejsce na dysku, żeby łączyć się z bazą bez interaktywnego pytania o hasło i bez trzymania go jawnym tekstem w treści skryptu.

Częste błędy

Podanie hasła flagą -p w skrypcie widocznym dla innych userów systemowych. Tak jak w każdym innym silniku, argument wywołania procesu jest widoczny w liście procesów systemu operacyjnego dla innych kont — `hdbuserstore` istnieje właśnie po to, żeby tego uniknąć w skryptach automatyzujących.

Połączenie z SYSTEMDB tam, gdzie potrzebny jest tenant. Zapytania do widoków `M_*` na System DB pokazują stan całej instalacji, nie danych konkretnego tenanta — próba odpytania tabeli aplikacji z poziomu System DB kończy się błędem, że taki obiekt nie istnieje, mimo że w tenantcie istnieje naprawdę.

Powiązane: Architektura SAP HANA: instancja, numer, nazwa SID, procesy · sapcontrol i zarządzanie instancją · Diagnostyka i miejsce na dysku

Źródła: dokumentacja SAP HANA, „SAP HANA SQL Command Network Protocol Reference" i „SAP HANA Administration Guide", rozdział „hdbuserstore"

Revision #42
Created 2026-09-25 00:01:07 UTC by Tomasz Konieczny
Updated 2026-10-26 14:08:34 UTC by Tomasz Konieczny