Scheduler i zadania w bazie

Poziom: podstawy **Dotyczy: Oracle Database 19c** **Czas: ~15 min**

Zadanie, które przestało się uruchamiać, zwykle nie ma nic wspólnego z samym zadaniem — winny jest koordynator schedulera, który trzeba zrestartować, albo parametr `JOB_QUEUE_PROCESSES` ustawiony na zero.

Fiszka

TYLKO ODCZYT

Sprawdzenie listy zadań schedulera

```
SELECT job_name, state, enabled, last_start_date, next_run_date
FROM dba_scheduler_jobs
ORDER BY job_name;
```

Przykład:

```
JOB_NAME                       STATE           ENABLED
LAST_START_DATE                   NEXT_RUN_DATE
------------------------------ --------------- -------
---------------------------------- ----------------------------------
JOB_EKSPORT_RAPORTOW             SCHEDULED       TRUE    25-SEP-26 02.00.11.000000 AM
+02:00 26-SEP-26 02.00.00.000000 AM +02:00
JOB_CZYSZCZENIE_LOGOW            RUNNING         TRUE    25-SEP-26 08.00.02.000000 AM
+02:00 26-SEP-26 08.00.00.000000 AM +02:00
```

TYLKO ODCZYT

Sprawdzenie zadań aktualnie uruchomionych

```
SELECT job_name, session_id, running_instance, elapsed_time
FROM dba_scheduler_running_jobs;
```

Przykład:

```
JOB_NAME                       SESSION_ID RUNNING_INSTANCE ELAPSED_TIME
------------------------------ ---------- ---------------- ---------------
JOB_CZYSZCZENIE_LOGOW                 305                1 +00 00:12:34.0
```

TYLKO ODCZYT

Sprawdzenie czasu działania konkretnego joba przez v$session_longops

```
SELECT s.sid, s.serial#, s.username, l.elapsed_seconds, l.sofar, l.totalwork
FROM v$session s, dba_scheduler_running_jobs j, v$session_longops l
WHERE s.sid = j.session_id
  AND s.sid = l.sid
  AND j.job_name = 'NazwaJobu';
```

Łączymy trzy widoki naraz, żeby dla konkretnej, znanej z góry nazwy joba dostać nie tylko fakt, że działa, ale i szczegółowy postęp z `v$session_longops`, opisanego szerzej na stronie Sesje: aktywne, nieaktywne, najdłuższe.

Przykład:

```
       SID    SERIAL# USERNAME             ELAPSED_SECONDS      SOFAR  TOTALWORK
---------- ---------- -------------------- --------------- ---------- ----------
       305         12 SYSTEM                         1284      84200     120000
```

TYLKO ODCZYT

Zatrzymanie zadania: DBMS_SCHEDULER.STOP_JOB

```
EXEC DBMS_SCHEDULER.STOP_JOB('NazwaJobu');
```

Przerywa działające zadanie i ustawia jego stan na `STOPPED` — job pozostaje zdefiniowany, tylko przestaje się wykonywać do czasu ponownego uruchomienia albo najbliższego uruchomienia wg harmonogramu.

Przykład:

```
SQL> EXEC DBMS_SCHEDULER.STOP_JOB('JOB_CZYSZCZENIE_LOGOW');

PL/SQL procedure successfully completed.
```

□□ **exec** = zastąp bieżący proces powłoki wskazanym programem, bez tworzenia nowego procesu

□□ **TYLKO ODCZYT**

Uruchomienie zadania ręcznie: DBMS_SCHEDULER.RUN_JOB

```
EXEC DBMS_SCHEDULER.RUN_JOB('NazwaJobu');
```

Przykład:

```
SQL> EXEC DBMS_SCHEDULER.RUN_JOB('JOB_CZYSZCZENIE_LOGOW');

PL/SQL procedure successfully completed.
```

□□ **exec** = zastąp bieżący proces powłoki wskazanym programem, bez tworzenia nowego procesu

□□ **TYLKO ODCZYT**

Wyłączenie schedulera: SCHEDULER_DISABLED na TRUE

```
EXEC DBMS_SCHEDULER.SET_SCHEDULER_ATTRIBUTE('SCHEDULER_DISABLED', 'TRUE');
```

Przykład:

```
SQL> EXEC DBMS_SCHEDULER.SET_SCHEDULER_ATTRIBUTE('SCHEDULER_DISABLED', 'TRUE');

PL/SQL procedure successfully completed.
```

□□ **exec** = zastąp bieżący proces powłoki wskazanym programem, bez tworzenia nowego procesu

□□ **TYLKO ODCZYT**

Włączenie schedulera: SCHEDULER_DISABLED na FALSE

```
EXEC DBMS_SCHEDULER.SET_SCHEDULER_ATTRIBUTE('SCHEDULER_DISABLED', 'FALSE');
```

Przykład:

```
SQL> EXEC DBMS_SCHEDULER.SET_SCHEDULER_ATTRIBUTE('SCHEDULER_DISABLED', 'FALSE');


PL/SQL procedure successfully completed.
```

exec = zastąp bieżący proces powłoki wskazanym programem, bez tworzenia nowego procesu

ZMIENIA SYSTEM

Restart koordynatora: wyzerowanie JOB_QUEUE_PROCESSES

```
ALTER SYSTEM SET job_queue_processes = 0;
```

> **Pojęcie: JOB_QUEUE_PROCESSES.** Parametr instancji określający maksymalną liczbę procesów wykonujących zadania schedulera równolegle. Ustawienie go na `0` zatrzymuje cały mechanizm — żaden nowy job się nie uruchomi, dopóki nie przywrócimy wartości dodatniej.

Przykład:

```
SQL> ALTER SYSTEM SET job_queue_processes = 0;


System altered.
```

SET = Switch Embedded Teaming: zespół kart sieciowych zbudowany w przełączniku wirtualnym Hyper-V; łączy dwie karty w jedno łącze bez ustawień po stronie przełącznika fizycznego, więc bez LACP

ZMIENIA SYSTEM

Restart koordynatora: przywrócenie JOB_QUEUE_PROCESSES

```
ALTER SYSTEM SET job_queue_processes = 1000;
```

Powrót do wartości roboczej po chwili — sam ten cykl `0` → wartość docelowa wymusza, żeby proces koordynatora zakończył się i wystartował od nowa, co w praktyce jest jego restartem bez zatrzymywania całej instancji.

Przykład:

```
SQL> ALTER SYSTEM SET job_queue_processes = 1000;


System altered.
```

□□ **SET** = Switch Embedded Teaming: zespół kart sieciowych zbudowany w przełączniku wirtualnym Hyper-V; łączy dwie karty w jedno łącze bez ustawień po stronie przełącznika fizycznego, więc bez LACP

□□ **ZMIENIA SYSTEM**

Ustawienie trwałej wartości JOB_QUEUE_PROCESSES

```
ALTER SYSTEM SET job_queue_processes = 1000 SCOPE=BOTH;
```

Trwałą wartość tego parametru, przeżywającą restart instancji, ustawiamy z `SCOPE=BOTH`.

Przykład:

```
SQL> ALTER SYSTEM SET job_queue_processes = 1000 SCOPE=BOTH;


System altered.
```

□□ **SET** = Switch Embedded Teaming: zespół kart sieciowych zbudowany w przełączniku wirtualnym Hyper-V; łączy dwie karty w jedno łącze bez ustawień po stronie przełącznika fizycznego, więc bez LACP

□□ **TYLKO ODCZYT**

Dodatkowe wyłączenie starszych zadań DBMS_JOB

```
EXEC DBMS_IJOB.SET_ENABLED(FALSE);
```

Instancje z zadaniami zdefiniowanymi jeszcze starszym pakietem `DBMS_JOB` mają osobny wyłącznik na poziomie konkretnego zadania — `DBMS_IJOB.SET_ENABLED(FALSE)` wyłącza je, `TRUE` włącza z powrotem. Sięgamy po to tylko w bazach, które nadal mają odziedziczone zadania sprzed migracji na `DBMS_SCHEDULER`.

Przykład:

```
SQL> EXEC DBMS_IJOB.SET_ENABLED(FALSE);
```

```
PL/SQL procedure successfully completed.
```

exec = zastąp bieżący proces powłoki wskazanym programem, bez tworzenia nowego procesu

ZMIENIA SYSTEM

Wyczyszczenie shared pool przy zawieszonym schedulerze

```
ALTER SYSTEM FLUSH SHARED_POOL;
```

Krok pomocniczy, po który sięgamy, gdy restart koordynatora przez `JOB_QUEUE_PROCESSES` sam w sobie nie wystarcza — w shared pool, opisanym na stronie Architektura Oracle: instancja, baza, SGA, PGA, procesy tła, potrafi utkwić plan wykonania powiązany z zablokowanym zadaniem.

Przykład:

```
SQL> ALTER SYSTEM FLUSH SHARED_POOL;


System altered.
```

SGA = System Global Area: wspólna pamięć instancji Oracle · **PGA** = Program Global Area: prywatna pamięć procesu serwera Oracle

TYLKO ODCZYT

Sprawdzenie historii uruchomień joba

```
SELECT log_date, job_name, status, error#, run_duration
FROM dba_scheduler_job_run_details
WHERE job_name = 'NazwaJobu'
ORDER BY log_date DESC
FETCH FIRST 10 ROWS ONLY;
```

`error#` różny od zera wskazuje, że zadanie zakończyło się błędem — konkretny numer odpowiada standardowemu kodowi błędu Oracle (`ORA-nnnnn`), a `run_duration` pokazuje, ile faktycznie trwało wykonanie, niezależnie od tego, czy zakończyło się sukcesem.

Przykład:

```
LOG_DATE                              JOB_NAME              STATUS     ERROR# RUN_DURATION
------------------------------------- --------------------- ---------- ------ ------------
25-SEP-26 02.00.14.000000 AM +02:00 JOB_EKSPORT_RAPORTOW  SUCCEEDED       0 +00 00:04:22
24-SEP-26 02.00.09.000000 AM +02:00 JOB_EKSPORT_RAPORTOW  FAILED       1017 +00 00:00:02
```

Wiersz ze `STATUS = FAILED` i `ERROR# = 1017` mówi wprost, że przyczyną był `ORA-01017: invalid username/password` — zwykle skutek wygasłego hasła konta, pod którym zadanie się łączy, opisanego na stronie Użytkownicy, role i uprawnienia.

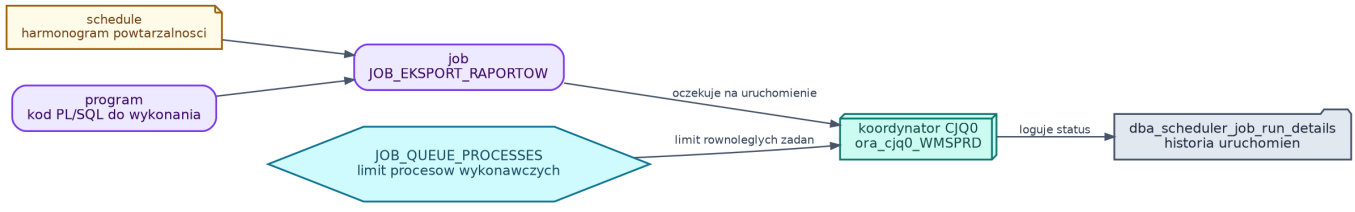
Z życia wzięte

Z ŻYCIA WZIĘTE

W poniedziałek rano Marek dzwoni, że nocny raport zapasów magazynowych z WMSPRD nie doszedł do jego skrzynki. Tomek sprawdza `dba_scheduler_jobs` i widzi, że `JOB_EKSPORT_RAPORTOW` ma status SCHEDULED, ale nigdy się nie uruchomił, bo ktoś wcześniej ustawił `JOB_QUEUE_PROCESSES` na zero przy diagnozowaniu innego problemu. Po przywróceniu wartości roboczej koordynator schedulera CJQ0 wstaje ponownie i zaległy raport odpala się ręcznie poleceniem `RUN_JOB`.

DBMS_SCHEDULER

Pojęcie: DBMS_SCHEDULER. Wbudowany pakiet PL/SQL do planowania i uruchamiania zadań wewnątrz bazy — następca starszego `DBMS_JOB`, wciąż obecnego w bazie dla kompatybilności wstecznej. Cztery pojęcia opisują jego model pracy:



Schedule i program razem tworzą job, job czeka na koordynatora CJQ0, a JOB_QUEUE_PROCESSES ogranicza liczbę zadań wykonywanych naraz (kliknięcie otwiera pełny

rozmiar)

> **Pojęcie: job.** Pojedyncze zadanie — co ma się wykonać (kod PL/SQL, procedura, polecenie systemowe) i kiedy.

> **Pojęcie: program.** Zdefiniowana osobno akcja do wykonania, wielokrotnego użytku — kilka jobów może odwoływać się do jednego programu z różnym harmonogramem.

> **Pojęcie: schedule.** Harmonogram — reguła powtarzalności, niezależna od konkretnego joba, również wielokrotnego użytku.

> **Pojęcie: window.** Okno czasowe z przypisanym poziomem priorytetu zasobów (przez Resource Manager), w którym zadania mogą się wykonywać z określonym limitem zasobów.

Restart schedulera programów

> **Pojęcie: SCHEDULER_DISABLED.** Atrybut schedulera, którym wyłączamy uruchamianie zadań całkowicie — inaczej niż zatrzymanie pojedynczego joba, ten wyłącznik dotyczy wszystkich zadań schedulera naraz.

> **Pojęcie: koordynator schedulera (CJQ0).** Proces tła odpowiedzialny za faktyczne odpalanie zaplanowanych zadań — widoczny w systemie jako `ora_cjq0_NazwaSID`. Gdy zadania przestają się uruchamiać mimo poprawnego harmonogramu i `SCHEDULER_DISABLED = FALSE`, podejrzanym jest właśnie ten proces, nie definicja samego joba. Restart koordynatora wymuszamy pośrednio, przez zmianę liczby procesów kolejki zadań.

Częste błędy

Zatrzymanie joba przez STOP_JOB i pozostawienie go tak na stałe. `STOP_JOB` nie zmienia harmonogramu — job wystartuje ponownie przy najbliższym zaplanowanym uruchomieniu, chyba że dodatkowo go wyłączymy przez `DBMS_SCHEDULER.DISABLE`.

Mylenie SCHEDULER_DISABLED z JOB_QUEUE_PROCESSES = 0. Pierwsze wyłącza scheduler na poziomie logicznym, drugie odbiera mu procesy do wykonywania zadań na poziomie instancji — oba dają podobny efekt widoczny z zewnątrz, ale to dwa różne przełączniki i warto sprawdzić oba, jeśli zadania nie startują.

Powiązane: Sesje: aktywne, nieaktywne, najdłuższe · Architektura Oracle: instancja, baza, SGA, PGA, procesy tła · Użytkownicy, role i uprawnienia

Źródła: notatki własne, dokumentacja Oracle Database PL/SQL Packages and Types Reference 19c

---

Revision #42
Created 2026-09-24 23:34:19 UTC by Tomasz Konieczny
Updated 2026-10-26 14:08:50 UTC by Tomasz Konieczny