Architektura i środowisko

Instancja kontra baza, procesy tła, SGA i PGA, zmienne środowiskowe, TNS i SQL*Plus.

- Oracle: baza, którą Larry Ellison sprzedał CIA, zanim ją napisał
- Architektura Oracle: instancja, baza, SGA, PGA, procesy tła
- Zmienne środowiskowe, TNS i logowanie do instancji
- SQL*Plus: klient i jego konfiguracja

Oracle: baza, którą Larry Ellison sprzedał CIA, zanim ją napisał

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

Ta strona otwiera książkę „Oracle DB": skąd wzięła się nazwa tego silnika, ile prawdy jest w krążącej o niej anegdocie i dlaczego licencjonowanie Oracle jest tematem, którego administrator nie może zignorować.

Skróty użyte na tej stronie (4)

Skrót	Co znaczy
CIA	Central Intelligence Agency: amerykańska agencja wywiadowcza, pierwszy klient komercyjnego Oracle
IBM	International Business Machines: firma, w której laboratorium powstał model relacyjny i pierwszy działający silnik SQL
SQL	Structured Query Language: język zapytań do relacyjnych baz danych
WMS	Warehouse Management System: system zarządzania magazynem

Skąd się wzięło

W 1970 roku Edgar Codd opublikował w laboratorium IBM artykuł opisujący model relacyjny, ale sam IBM nie spieszył się z komercyjnym produktem opartym na własnym pomyśle. Larry Ellison, pracujący wtedy w firmie Ampex, przeczytał ten artykuł i uznał go za rzecz oczywistą, a jednocześnie niezbudowaną przez nikogo. W 1977 roku razem z Bobem Minerem i Edem Oatesem założył w Santa Clara firmę Software Development Laboratories, z kapitałem startowym rzędu dwóch tysięcy dolarów.

Nazwa Oracle nie wzięła się z marketingu, tylko z projektu, przy którym Ellison, Miner i Oates wcześniej razem pracowali dla CIA, gdzie nosił on kryptonim Oracle. Kiedy CIA potrzebowała systemu bazodanowego, trafiła właśnie do nich, a firma przyjęła nazwę tego zamówienia na własny

produkt. W tej części anegdota jest prawdziwa: CIA rzeczywiście była pierwszym klientem, a nazwa firmy pochodzi od nazwy jej projektu, nie odwrotnie.

Druga część, że Ellison „sprzedał" bazę danych, zanim ją napisał, jest prawdziwa tylko częściowo. Umowa z CIA rzeczywiście zapadła, zanim istniał gotowy produkt: w 1978 roku pierwsza wersja, pisana w asemblerze, była wciąż wewnętrznym prototypem, nie czymś, co dało się sprzedać drugi raz. Software Development Laboratories dopiero dostała za to zadanie zapłatę i zbudowała system, którego jeszcze nie miała gotowego na papierze. Nie jest to więc historia o sprzedaży fikcyjnego produktu, tylko o kontrakcie badawczym podpisanym przed końcem prac, co w branży oprogramowania rządowego nie było i nie jest niczym niezwykłym.

Pierwszy komercyjny silnik trafił na rynek w 1979 roku pod nazwą Oracle w wersji drugiej, bo założyciele uznali, że klienci niechętnie kupują produkt oznaczony jako pierwsza wersja. Był to pierwszy dostępny na rynku silnik relacyjny z obsługą SQL, wyprzedzający własny komercyjny produkt IBM. Firma zmieniała nazwę jeszcze dwukrotnie, aż w 1982 roku przyjęła na stałe nazwę swojego pierwszego produktu: Oracle.

Jak to działa w jednym obrazku

Zapytanie do Oracle przechodzi tę samą drogę niezależnie od tego, czy pyta o nią skaner w magazynie, czy raport księgowego: od aplikacji przez instancję do bazy danych zamontowanej na dysku.



Skanery magazynowe wysyłają zapytania przez WMS do instancji Oracle na ora01, która montuje i otwiera bazę WMSPRD zapisaną fizycznie jako datafile (kliknięcie otwiera pełny rozmiar)

Dlaczego Zielona

> **Z ŻYCIA WZIĘTE**
>
> O szóstej rano w Swarzędzu czterdzieści skanerów loguje się naraz do WMS, a WMS odpytuje instancję Oracle na `ora01` o stan magazynu. Marek już nie raz dzwonił, że skanery „muszą się rozgrzać", chociaż w rzeczywistości to instancja dopiero wtedy dostaje pierwszy poranny szczyt sesji po nocnej ciszy. Dopiero druga taka awaria nauczyła zespół, żeby sprawdzać sesje na bazie, zanim zaczną obwiniać sprzęt skanerów.

WMS, system magazynowy kupiony od zewnętrznego dostawcy, wymaga Oracle, więc na `ora01` stoi baza `WMSPRD` i nikt w Zielonej nie miał tu wyboru silnika. Licencjonowanie Oracle liczy rdzenie procesora, a nie liczbę użytkowników, więc zanim pan Zbigniew zgodzi się na dodanie mocy obliczeniowej do serwera, najpierw sprawdzamy, czy umowa licencyjna w ogóle na to pozwala. To odróżnia Oracle od PostgreSQL w tej samej serwerowni: koszt złej decyzji administracyjnej tutaj bywa liczony w tysiącach złotych za rdzeń, nie tylko w czasie przestoju.

Mapa tej książki

Książka „Oracle DB" składa się z siedmiu rozdziałów rozłożonych na dwa katalogi plików, budowanych na słowniku z książki „Bazy danych: fundamenty".

Rozdział	Na jakie pytanie odpowiada
Oracle: baza, którą Larry Ellison sprzedał CIA, zanim ją napisał	Skąd wzięła się nazwa Oracle i dlaczego licencjonowanie jest tematem administratora
Architektura i środowisko	Czym różni się instancja od bazy, czym jest SGA i PGA, jak zalogować się przez SQL*Plus
Rozpoznanie i diagnostyka	Jak sprawdzić wersję, parametry, sesje, blokady i zadania schedulera
Użytkownicy i uprawnienia	Jak zarządzać rolami, kontem monitorującym i łączami bazodanowymi
Przestrzeń i pliki bazy	Czym jest tablespace, datafile, UNDO, TEMP, redo log i plik kontrolny
Backup i odtwarzanie	Tryb ARCHIVELOG, RMAN, Data Pump i odtworzenie z backupu Networkera
Rozmiary i wydajność	Jak sprawdzić rozmiary tabel, segmentów i indeksów
Procedury	Start i stop instancji, utworzenie nowej bazy, upgrade silnika, migracja na inny serwer

Powiązane: Baza danych, silnik, instancja i schemat · Architektura Oracle, instancja, baza, SGA, PGA, procesy tła · Użytkownicy, role i uprawnienia

Źródła: Software Development Laboratories, historia firmy, content.dsp.co.uk „The History of Oracle: 1977-2000" · stacksync.com „The Paper Nobody Acted on: The Origin Story of Oracle" · gizmodo.com „Larry Ellison's Oracle Started As a CIA Project" · Wikipedia, „Oracle Corporation"

Architektura Oracle: instancja, baza, SGA, PGA, procesy tła

Poziom: podstawy **Dotyczy: Oracle Database 19c** **Czas: ~20 min**

Zanim zaczniemy cokolwiek diagnozować w Oracle, musimy mieć jasność co do jednego rozróżnienia: instancja i baza danych to dwie osobne rzeczy, które da się nawet uruchamiać i zatrzymywać niezależnie od siebie.

Skróty użyte na tej stronie (11)

Skrót	Co znaczy
AMM	Automatic Memory Management: tryb Oracle, w którym silnik sam rozdziela całą pamięć między SGA i PGA
ASMM	Automatic Shared Memory Management: tryb Oracle, w którym silnik sam rozdziela pamięć wewnątrz SGA, ale granica między SGA i PGA jest ustawiona na sztywno
CDB	Container Database: baza-kontener w Oracle 12c i nowszych, trzyma w sobie bazy PDB
GB	Gigabyte: miliard bajtów, jednostka pojemności pamięci i plików, tu najczęściej rozmiar modelu albo pamięci karty graficznej
PDB	Pluggable Database: baza wpinana do kontenera CDB w Oracle
PGA	Program Global Area: prywatna pamięć procesu serwera Oracle
RAM	Random Access Memory: pamięć operacyjna
RMAN	Recovery Manager: narzędzie Oracle do kopii zapasowych
SGA	System Global Area: wspólna pamięć instancji Oracle
SQL	Structured Query Language: język zapytań do relacyjnych baz danych
WMS	Warehouse Management System: system zarządzania magazynem

Fiszka

□□ ZMIENIA SYSTEM

Uruchomienie samej instancji: STARTUP NOMOUNT

```
STARTUP NOMOUNT;
```

Uruchamiamy samą instancję — czytamy plik parametrów, alokujemy SGA, startujemy procesy tła. Na tym etapie baza jeszcze nie jest znana instancji, bo nie przeczytaliśmy jeszcze pliku kontrolnego.

□□ **SGA** = System Global Area: wspólna pamięć instancji Oracle

□□ ZMIENIA SYSTEM

Montowanie bazy: ALTER DATABASE MOUNT

```
ALTER DATABASE MOUNT;
```

Instancja czyta plik kontrolny i poznaje lokalizacje datafile'i oraz redo logów — baza jest zamontowana, ale jeszcze zamknięta na czytanie i zapis.

□□ ZMIENIA SYSTEM

Otwarcie bazy: ALTER DATABASE OPEN

```
ALTER DATABASE OPEN;
```

Instancja otwiera datafile'e do odczytu i zapisu. Dopiero teraz baza jest dostępna dla zwykłych sesji.

Przykład:

```
SQL> STARTUP NOMOUNT;
ORACLE instance started.

Total System Global Area 7180646608 bytes
Fixed Size                  8956112 bytes
Variable Size            1191182336 bytes
Database Buffers         5972688896 bytes
Redo Buffers                7819264 bytes
```

```
SQL> ALTER DATABASE MOUNT;

Database altered.

SQL> ALTER DATABASE OPEN;

Database altered.
```

⚠ ZMIENIA SYSTEM

Uruchomienie instancji jedną komendą: STARTUP

```
STARTUP;
```

W praktyce najczęściej powyższe trzy kroki zwijamy w jedną komendę.

Przykład:

```
SQL> STARTUP;
ORACLE instance started.

Total System Global Area 7180646608 bytes
Fixed Size                  8956112 bytes
Variable Size            1191182336 bytes
Database Buffers         5972688896 bytes
Redo Buffers                7819264 bytes
Database mounted.
Database opened.
```

✓ TYLKO ODCZYT

Sprawdzenie aktualnych wartości SGA i PGA: SHOW PARAMETER

```
SHOW PARAMETER sga_target
```

Wyświetlamy aktualną wartość parametru bez przeszukiwania całego `v$parameter` ręcznie — komenda `SHOW PARAMETER` działa tylko w SQL*Plus, opisana szerzej na stronie SQL*Plus: klient i jego konfiguracja.

Przykład:

```
NAME                                 TYPE        VALUE
------------------------------------ ----------- ------------------------------
sga_target                           big integer 45G
```

SGA = System Global Area: wspólna pamięć instancji Oracle · **PGA** = Program Global Area: prywatna pamięć procesu serwera Oracle · **SQL** = Structured Query Language: język zapytań do relacyjnych baz danych · **show** = FortiOS: wypisuje konfigurację, w podstawowej postaci tylko ustawienia różniące się od domyślnych, a z wariantem full-configuration także wartości domyślne

ZMIENIA SYSTEM

Ustawienie wartości docelowej: sga_target

```
ALTER SYSTEM SET sga_target = 45G SCOPE=SPFILE;
```

`SCOPE=SPFILE` zapisuje zmianę tylko do pliku parametrów — instancja działająca teraz jej nie widzi, potrzebny jest restart; pełne wyjaśnienie parametru `SCOPE` jest niżej.

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 maksymalnego rozmiaru SGA: sga_max_size

```
ALTER SYSTEM SET sga_max_size = 45G SCOPE=SPFILE;
```

`sga_max_size` wyznacza sufit, do którego `sga_target` może rosnąć bez restartu instancji — samo podniesienie `sga_target` powyżej `sga_max_size` się nie powiedzie. Trzymamy się zasady `sga_target <= sga_max_size`; najprościej ustawić obie wartości równo.

SGA = System Global Area: wspólna pamięć instancji Oracle · **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 celu dla PGA: pga_aggregate_target

```
ALTER SYSTEM SET pga_aggregate_target = 11G SCOPE=SPFILE;
```

`pga_aggregate_target` to cel, do którego optymalizator dąży przy rozdzielaniu PGA między sesje — instancja może go chwilowo przekroczyć pod obciążeniem.

PGA = Program Global Area: prywatna pamięć procesu serwera Oracle · **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 twardego limitu PGA: pga_aggregate_limit

```
ALTER SYSTEM SET pga_aggregate_limit = 22G SCOPE=SPFILE;
```

`pga_aggregate_limit` to twardy sufit: instancja zaczyna zabijać albo ograniczać najbardziej żarłoczne sesje, gdy sumaryczne PGA go przekracza. Zasada wyjściowa: `pga_aggregate_limit` ustawiamy na dwukrotność `pga_aggregate_target`, ale nie mniej niż 6 GB.

PGA = Program Global Area: prywatna pamięć procesu serwera Oracle · **GB** = Gigabyte: miliard bajtów, jednostka pojemności pamięci i plików, tu najczęściej rozmiar modelu albo pamięci karty graficznej · **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

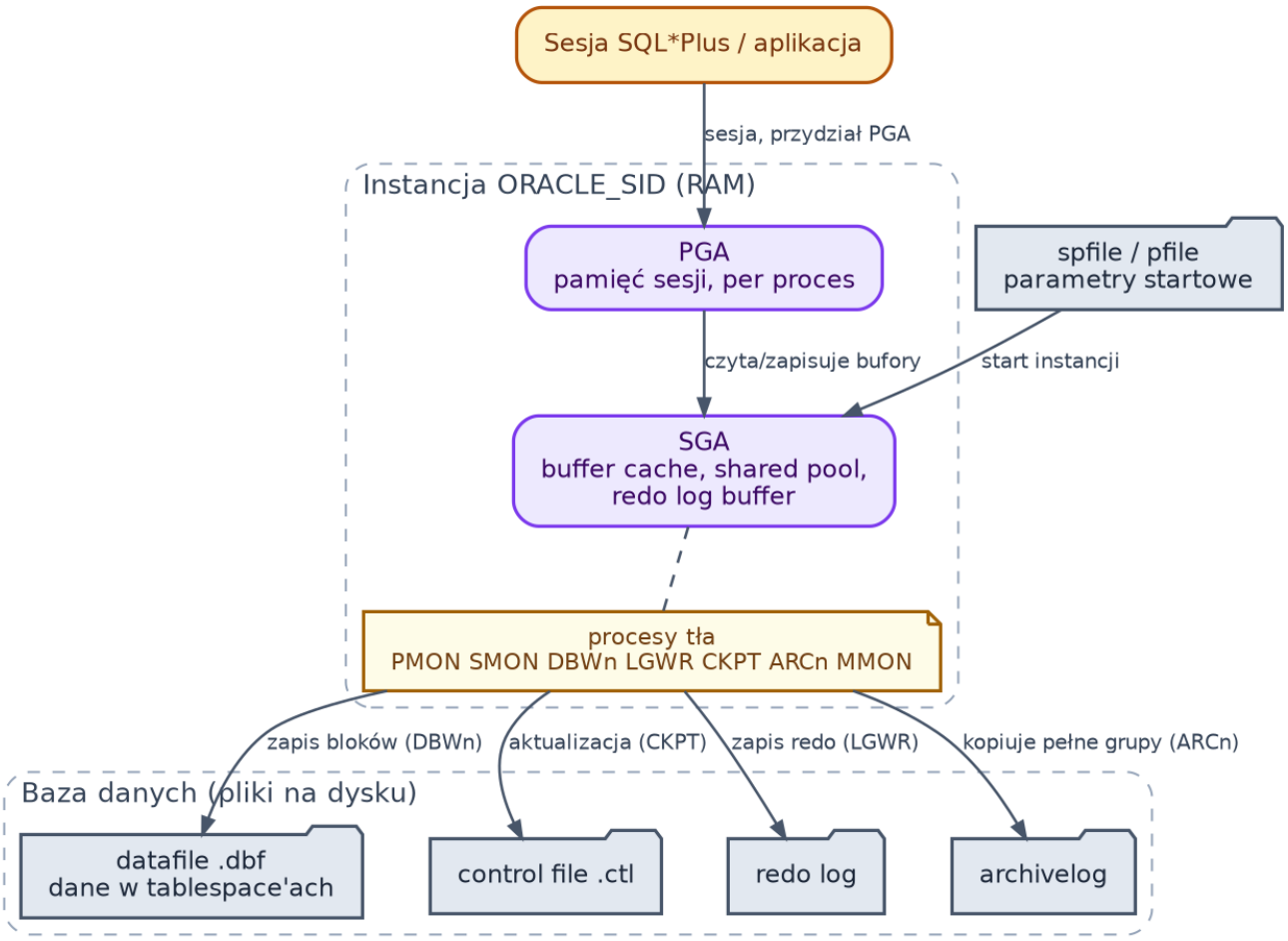
Z życia wzięte

Z ŻYCIA WZIĘTE

O godzinie 6:05 Marek dzwoni, że skanery w Swarzędzu wiszą, mimo że WMS na ora01 wygląda na uruchomiony. Tomek sprawdza `v$instance` i widzi, że instancja stoi w stanie MOUNT, bo poprzedniej nocy ktoś zatrzymał sekwencję startową w połowie. Dopiero po `ALTER DATABASE OPEN` baza WMSPRD otwiera pliki danych i skanery zaczynają logować zamówienia.

Instancja kontra baza danych

Pojęcie: instancja. Zbiór struktur pamięciowych w RAM-ie (SGA) razem z procesami tła, które nimi zarządzają. Instancja powstaje w momencie `startup` i znika w momencie `shutdown` — nic z tego, co w niej jest, nie przeżywa restartu serwera. Instancja ma swoją nazwę, `ORACLE_SID`, i to ją mamy na myśli, mówiąc „logujemy się do instancji WMSPRD".



Instancja w pamięci RAM (SGA, PGA i procesy tła) obsługuje sesję klienta i przez procesy tła zapisuje dane do plików bazy na dysku (kliknięcie otwiera pełny rozmiar)

Pojęcie: baza danych. Zbiór plików na dysku: pliki danych (datafile), pliki kontrolne (control file), pliki redo log i — jeśli tryb archiwizacji jest włączony — archiwalne kopie redo logów. Baza istnieje niezależnie od tego, czy jakakolwiek instancja jest akurat uruchomiona. Możemy mieć bazę danych na dysku i żadnej działającej instancji, która by ją obsługiwała.

Rozróżnienie ma znaczenie praktyczne, nie tylko akademickie — sekwencja startowa Oracle dosłownie przechodzi przez te dwa byty osobno, jak widać w komendach na górze strony.

`SHUTDOWN IMMEDIATE` odwraca kolejność — zamyka bazę, odmontowuje ją, a na końcu wyłącza instancję. Cały proces potrafi zająć nawet kilka minut, bo instancja musi domknąć wszystkie transakcje i sesje w toku, zanim faktycznie się zatrzyma.

SGA — pamięć współdzielona instancji

> **Pojęcie: SGA (System Global Area).** Obszar pamięci współdzielonej alokowany raz, przy starcie instancji, i widoczny dla wszystkich procesów tej instancji. To właśnie SGA widzimy w komunikacie startowym wyżej — cztery pozycje sumujące się na `Total System Global Area`.

`Fixed Size` to niewielki, stały obszar ze strukturami samego silnika — nie konfigurujemy go ręcznie. Reszta dzieli się na trzy właściwe komponenty:

> **Pojęcie: buffer cache.** Bufor bloków danych przeczytanych z datafile'i. Zapytanie, które trafia w blok już leżący w buffer cache, oszczędza kosztowny odczyt z dysku — to jest największy pojedynczy dźwigniowy element wydajności instancji, obok samego planu wykonania, o którym mówi strona Plan wykonania: jak go wywołać i jak go czytać.

> **Pojęcie: shared pool.** Bufor sparsowanych planów wykonania SQL, skompilowanego kodu PL/SQL i słownika danych. Dzięki niemu identyczne zapytanie wykonane drugi raz nie przechodzi ponownie przez parsowanie i optymalizację — silnik znajduje gotowy plan w shared pool. `alter system flush shared_pool;` czyści tę pamięć podręczną do zera — sięgamy po to rzadko, zwykle przy podejrzeniu, że w shared pool utkwił zły plan wykonania albo zawisł proces schedulera.

> **Pojęcie: redo log buffer.** Mały bufor, do którego procesy zapisujące trafiają zmiany danych, zanim trafią one do plików redo log na dysku. Proces LGWR opisany niżej regularnie go opróżnia.

Czwarty, opcjonalny składnik to large pool — dodatkowa pula pamięci odciążająca shared pool przy operacjach RMAN, sesjach współdzielonego serwera (shared server) i operacjach równoległych. Konfigurujemy go osobnym parametrem `LARGE_POOL_SIZE`, gdy shared pool zaczyna się fragmentować pod obciążeniem backupów.

PGA — pamięć prywatna procesu

> **Pojęcie: PGA (Program Global Area).** W odróżnieniu od SGA, PGA nie jest współdzielona — każdy proces serwerowy (każda sesja) dostaje własny obszar PGA na sortowanie, hashowanie w złączeniach hashowych i inne operacje, które nie mieszczą się w jednym przebiegu przez dane. Sesja, która wykonuje ciężki `ORDER BY` albo `GROUP BY` na dużym zbiorze, zużywa PGA proporcjonalnie do rozmiaru sortowanego zbioru.

Zarządzanie pamięcią: SGA & PGA Management

Oracle od wersji 9i potrafi zarządzać rozmiarem SGA i PGA automatycznie — nie musimy ręcznie ustawiać rozmiaru każdego bufora z osobna. Trzy tryby automatycznego zarządzania pamięcią różnią się tym, co dokładnie automatyzują:

Tryb	Parametr	Co robi
AMM (Automatic Memory Management)	`MEMORY_TARGET`, `MEMORY_MAX_TARGET`	Silnik sam dzieli całą pamięć instancji między SGA i PGA
ASMM (Automatic Shared Memory Management)	`SGA_TARGET`, `PGA_AGGREGATE_TARGET`	Rozmiary SGA i PGA ustawiamy osobno, wewnętrzny podział komponentów SGA jest automatyczny
Ręczny	`DB_CACHE_SIZE`, `SHARED_POOL_SIZE`, ...	Każdy komponent SGA ustawiamy z osobna

W praktyce na serwerach produkcyjnych najczęściej pracujemy w trybie ASMM — dajemy więcej kontroli nad tym, ile pamięci dostaje SGA, a ile PGA, niż przy pełnym AMM.

> **Pojęcie: SCOPE w ALTER SYSTEM.** `SCOPE=MEMORY` zmienia parametr tylko w działającej instancji, do najbliższego restartu. `SCOPE=SPFILE` zapisuje zmianę tylko do pliku parametrów, ale instancja działająca teraz jej nie widzi — potrzebny jest restart. `SCOPE=BOTH` robi obie rzeczy naraz, o ile parametr da się zmienić w locie. Parametry `sga_max_size` i `pga_aggregate_limit` należą do tych, które wymagają `SCOPE=SPFILE` i restartu instancji — parametry z przyrostkiem `_target` (`sga_target`, `pga_aggregate_target`) działają natomiast od razu.

Punktem wyjścia do wyliczenia wszystkich czterech wartości jest ilość pamięci RAM serwera:

```
pga_aggregate_target = (RAM_serwera x 0,8) x 0,2
sga_max_size         = (RAM_serwera x 0,8) x 0,8
sga_target           = sga_max_size
pga_aggregate_limit  = pga_aggregate_target x 2  (minimum 6G)
```

Dla serwera z 70 GB RAM-u wychodzi z tego `pga_aggregate_target` = 11 GB, `sga_max_size` = `sga_target` = 45 GB, `pga_aggregate_limit` = 22 GB. To są wartości wyjściowe, nie docelowe — dostrojenie SGA i PGA do konkretnego obciążenia wymaga już obserwacji rzeczywistego zużycia, nie samej proporcji do RAM-u serwera.

Procesy tła

> **Pojęcie: proces tła.** Osobny proces systemu operacyjnego, uruchamiany razem z instancją, który wykonuje jedno konkretne zadanie w tle — bez udziału sesji użytkownika. Widzimy je w systemie jako osobne procesy `ora_xxxx_NazwaSID`. Siedem z nich pojawia się praktycznie w każdej instancji:

Proces	Zadanie
PMON (Process Monitor)	sprząta po awarii procesu sesji — zwalnia blokady i pamięć po sesji, która padła
SMON (System Monitor)	odtwarza instancję po awarii przy starcie, sprząta tymczasowe segmenty
DBWn (Database Writer)	zapisuje zmodyfikowane bloki z buffer cache do datafile'i; może działać w kilku kopiach (DBW0, DBW1...)
LGWR (Log Writer)	zapisuje zawartość redo log buffer do plików redo log na dysku, przy każdym commit
CKPT (Checkpoint)	uruchamia checkpoint — punkt, w którym instancja aktualizuje pliki kontrolne i nagłówki datafile'i o to, które dane są już bezpiecznie zapisane
ARCn (Archiver)	kopiuje zapełnione grupy redo logów do archive logów, gdy baza pracuje w trybie `ARCHIVELOG`
MMON (Manageable Monitor)	zbiera statystyki wydajnościowe (AWR), generuje alerty progowe

DBWn i LGWR to para, która najczęściej wraca w rozmowach o wydajności zapisu: LGWR pisze redo log synchronicznie przy każdym commit, więc to jego opóźnienie najbardziej odczuwa aplikacja transakcyjna, natomiast DBWn zapisuje bloki danych asynchronicznie, w tle, i to jego szybkość decyduje o długości checkpointu.

Pliki bazy danych

Baza danych to, jak ustaliliśmy wyżej, zbiór plików na dysku. Cztery rodzaje plików pojawiają się w każdej instalacji, piąty tylko przy pracy z parametrami:

> **Pojęcie: tablespace.** Logiczny kontener grupujący jeden lub więcej plików danych (datafile), do którego przypisujemy tabele i indeksy przy ich tworzeniu. To warstwa pośrednia między bazą danych a pojedynczym plikiem na dysku, dzięki której możemy rozłożyć dane po kilku dyskach bez zmiany ani jednego zapytania. Szerzej: Baza danych, silnik, instancja i schemat.

Plik	Rola	Szczegóły
datafile (`.dbf`)	przechowuje właściwe dane tabel i indeksów, pogrupowane w tablespace	Tablespace — segmenty, extenty, datafile
control file (`.ctl`)	mapa bazy — lista datafile'i, redo logów, numer SCN, historia backupów	Control file — budowa i odtwarzanie
redo log	zapis każdej zmiany danych, zanim trafi ona trwale do datafile'a	Redo log — grupy, przełączanie, wielkość, Log transakcyjny: WAL, redo, transaction log
archivelog	trwała kopia zapełnionej grupy redo logów, podstawa odtwarzania do punktu w czasie	Tryb ARCHIVELOG i archiwizacja
spfile / pfile	plik parametrów startowych instancji	patrz niżej

> **Pojęcie: spfile i pfile.** `pfile` (parameter file) to zwykły plik tekstowy z parametrami instancji, edytowalny ręcznie, wczytywany raz przy starcie. `spfile` (server parameter file) to jego binarny następca, który silnik potrafi modyfikować sam poleceniem `ALTER SYSTEM ... SCOPE=SPFILE` — dzięki temu zmiana parametru nie wymaga ręcznej edycji pliku i restartu z linii poleceń. Sprawdzenie, z którego z nich aktualnie wystartowała instancja, opisujemy na stronie Rozpoznanie instancji: wersja, parametry, komponenty.

Baza kontenerowa — CDB i PDB

Od wersji 12c Oracle wprowadził architekturę wielodzierżawczą (multitenant), w której jedna instancja obsługuje nie jedną bazę, tylko kilka.

> **Pojęcie: CDB (Container Database).** Baza kontenerowa — nadrzędny kontener, który trzyma wspólny słownik danych, wspólne pliki kontrolne i redo logi dla wszystkich baz osadzonych w nim. Sam CDB nazywa się `CDB$ROOT` i jest zawsze pierwszym kontenerem, do którego trafiamy po zalogowaniu.

> **Pojęcie: PDB (Pluggable Database).** Baza podłączalna — właściwa baza aplikacyjna, osadzona wewnątrz CDB. Ma własny zestaw tablespace'ów i własnych użytkowników, ale dzieli procesy tła i SGA z całą instancją CDB. Jedna instancja Oracle może hostować wiele PDB-ów jednocześnie, każdy odizolowany od pozostałych na poziomie danych.

> **Pojęcie: kontener seed.** `PDB$SEED` to szablon, z którego tworzymy nowe PDB — sam nigdy nie służy do przechowywania danych aplikacji i pozostaje zwykle zamknięty na `READ ONLY`.

Przełączanie się między CDB$ROOT a poszczególnymi PDB-ami z poziomu sesji SQL*Plus opisujemy na stronie Zmienne środowiskowe, TNS i logowanie do instancji — od 21c to `ALTER SESSION SET CONTAINER` zastępuje dawne przełączanie instancji poleceniem `oraenv`.

Prefiks C##. Użytkownik utworzony na poziomie `CDB$ROOT`, widoczny i uprawniony potencjalnie we wszystkich PDB-ach naraz, musi mieć nazwę zaczynającą się od `C##` — to wymóg silnika, nie konwencja. Konto monitorujące Zabbixa w architekturze kontenerowej nazywa się dlatego `C##ZABBIX`, a nie `ZABBIX` — pełną procedurę jego zakładania opisuje strona Konto monitorujące dla Zabbixa.

Częste błędy

Zmiana `sga_max_size` albo `pga_aggregate_limit` przez `SCOPE=MEMORY` lub bez podania scope. Domyślny `SCOPE` w `ALTER SYSTEM` to `BOTH`, ale oba te parametry i tak nie dają się zmienić w locie — komenda zwróci błąd `ORA-02095: specified initialization parameter cannot be modified`, jeśli spróbujemy bez `SCOPE=SPFILE`, i będzie wymagać restartu instancji, żeby zmiana faktycznie zadziałała.

Ustawienie `sga_target` wyższego niż `sga_max_size`. Instancja odrzuci taką zmianę, dopóki najpierw nie podniesiemy `sga_max_size` — a to wymaga restartu.

Traktowanie CDB i zwykłej instancji tak samo. Zapytanie napisane pod klasyczną, niekontenerową instancję (`DBA_USERS`, `DBA_TABLESPACES`) w architekturze CDB pokazuje dane tylko bieżącego kontenera. Odpowiedniki `CDB_USERS`, `CDB_TABLESPACES` widzą dane wszystkich PDB-ów naraz, z dodatkową kolumną `CON_ID` mówiącą, do którego kontenera wiersz należy.

Powiązane: Zmienne środowiskowe, TNS i logowanie do instancji · Rozpoznanie instancji: wersja, parametry, komponenty · Log transakcyjny: WAL, redo, transaction log · Konta, role i uprawnienia: GRANT, REVOKE, zasada najmniejszych uprawnień

Źródła: notatki własne, dokumentacja Oracle Database 19c — „Oracle Database Concepts", „Oracle Database Administrator's Guide"

Zmienne środowiskowe, TNS i logowanie do instancji

Poziom: podstawy **Dotyczy: Oracle Database 19c, Linux** **Czas: ~20 min**

Zanim `sqlplus` w ogóle zapyta nas o hasło, powłoka musi wiedzieć, do którego katalogu instalacji i do której instancji się odnosimy — i to jest źródło większości pomyłek przy pracy na serwerze z kilkoma bazami naraz.

Skróty użyte na tej stronie (5)

Skrót	Co znaczy
CDB	Container Database: baza-kontener w Oracle 12c i nowszych, trzyma w sobie bazy PDB
RMAN	Recovery Manager: narzędzie Oracle do kopii zapasowych
SID	w Oracle: System Identifier, nazwa instancji; w Windows: Security Identifier, numer konta
SQL	Structured Query Language: język zapytań do relacyjnych baz danych
WMS	Warehouse Management System: system zarządzania magazynem

Fiszka

ZMIENIA SYSTEM

Zmiana domyślnego formatu daty sesji: NLS_DATE_FORMAT

```
export NLS_DATE_FORMAT="DD-MM-YYYY HH24:MI:SS"
```

Bywa wygodnie nadpisać sam format daty punktowo, bez dotykania reszty ustawień regionalnych ustawianych przez `NLS_LANG` — przydaje się to zwłaszcza przy pracy z RMAN-em i logami. Ta zmienna sprawia, że każda data zwrócona przez zapytanie albo wypisana przez RMAN pojawia się w jednym, przewidywalnym formacie, niezależnie od ustawień regionalnych systemu operacyjnego.

RMAN = Recovery Manager: narzędzie Oracle do kopii zapasowych · **export** = ustaw zmienną środowiskową widoczną dla uruchamianych programów

TYLKO ODCZYT

Zmiana instancji, do której logujemy się jako sysdba: oraenv

```
. oraenv
```

Uruchamiamy skrypt w bieżącej powłoce konta `oracle`. Zapyta o SID instancji, na którą chcemy się przełączyć — domyślna odpowiedź w nawiasach kwadratowych to instancja ustawiona obecnie.

Przykład:

```
oracle@ora01:~$ . oraenv
ORACLE_SID = [WMSPRD] ? WMSTEST
The Oracle base remains unchanged with value /oracle/db
oracle@ora01:~$ echo $ORACLE_SID
WMSTEST
```

SID = w Oracle: System Identifier, nazwa instancji; w Windows: Security Identifier, numer konta

TYLKO ODCZYT

Zalogowanie się jako sysdba po zmianie instancji: sqlplus

```
sqlplus / as sysdba
```

`sqlplus` bez podania danych logowania, z opcją `as sysdba`, uwierzytelnia nas przez system operacyjny — działa tylko wtedy, gdy jesteśmy zalogowani na konto `oracle` (albo konto należące do grupy `dba`) i gdy `ORACLE_SID` wskazuje instancję, do której chcemy się dostać.

Przykład:

```
oracle@ora01:~$ sqlplus / as sysdba

SQL*Plus: Release 19.0.0.0.0 - Production on Thu Sep 25 08:12:03 2026
Copyright (c) 1982, 2024, Oracle.  All rights reserved.

Connected to:
```

```
Oracle Database 19c Standard Edition 2 Release 19.22.0.0.0

SYS@WMSTEST: SQL> SELECT sys_context('USERENV','INSTANCE_NAME') AS INSTANCE FROM dual;

INSTANCE
----------------------------
WMSTEST
```

Zapytanie na końcu przykładu to zwykły nawyk kontrolny — po każdej zmianie instancji warto się upewnić, że prompt `SYS@WMSTEST` naprawdę odpowiada instancji, na której zamierzamy pracować, zanim wykonamy cokolwiek nieodwracalnego.

sqlplus = SQL*Plus: klient bazy Oracle

TYLKO ODCZYT

Przełączenie na inną instancję na tym samym serwerze: oraenv

```
. oraenv
```

Ten sam serwer często hostuje więcej niż jedną instancję. Powtórne wykonanie `oraenv` z inną odpowiedzią przełącza cały kontekst powłoki na drugą bazę.

Przykład:

```
oracle@ora01:~$ . oraenv
ORACLE_SID = [WMSTEST] ? WMSPRD
The Oracle base remains unchanged with value /oracle/db
oracle@ora01:~$ sqlplus / as sysdba

SQL*Plus: Release 19.0.0.0.0 - Production on Thu Sep 25 08:14:51 2026

Connected to:
Oracle Database 19c Standard Edition 2 Release 19.22.0.0.0

SYS@WMSPRD: SQL>
```

ZMIENIA SYSTEM

Przełączenie kontenera w architekturze CDB: ALTER SESSION SET CONTAINER

```
ALTER SESSION SET CONTAINER = NazwaPDB;
```

Od Oracle 21c, w architekturze bazy kontenerowej opisanej na stronie Architektura Oracle: instancja, baza, SGA, PGA, procesy tła, przełączanie między bazami osadzonymi w tej samej instancji CDB nie odbywa się już przez `oraenv`, tylko przez zmianę kontenera w ramach jednej sesji. Różnica jest istotna: `oraenv` zmienia instancję, do której łączy się cała powłoka, natomiast `ALTER SESSION SET CONTAINER` przełącza kontener w obrębie tej samej, już nawiązanej sesji do tej samej instancji CDB.

CDB = Container Database: baza-kontener w Oracle 12c i nowszych, trzyma w sobie bazy PDB · **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 · **SGA** = System Global Area: wspólna pamięć instancji Oracle · **PGA** = Program Global Area: prywatna pamięć procesu serwera Oracle

TYLKO ODCZYT

Logowanie lokalne jako sysdba: sqlplus

```
sqlplus / as sysdba
```

Opisane wyżej — uwierzytelnienie przez system operacyjny, bez hasła, tylko na koncie należącym do właściwej grupy systemowej.

sqlplus = SQL*Plus: klient bazy Oracle

TYLKO ODCZYT

Logowanie z podaniem użytkownika, hasła i aliasu TNS: sqlplus

```
sqlplus NazwaUsera/HasloUsera@NazwaInstancji
```

Standardowe logowanie sieciowe — `NazwaInstancji` musi być aliasem zdefiniowanym w `tnsnames.ora` widocznym z bieżącego `TNS_ADMIN`.

Przykład:

```
$ sqlplus app_user/********@WMSPRD

SQL*Plus: Release 19.0.0.0.0 - Production on Thu Sep 25 08:20:11 2026

Connected to:
Oracle Database 19c Standard Edition 2 Release 19.22.0.0.0

APP_USER@WMSPRD: SQL>
```

TNS = Transparent Network Substrate: warstwa sieciowa Oracle, przez którą klient znajduje i łączy się z listenerem bazy · **sqlplus** = SQL*Plus: klient bazy Oracle

TYLKO ODCZYT

Logowanie przez easy connect, bez wpisu w tnsnames.ora: sqlplus

```
sqlplus NazwaUsera/HasloUsera@NazwaSerwera:NumerPortu/NazwaUslugi
```

Easy connect pomija plik `tnsnames.ora` całkowicie — podajemy adres, port i nazwę usługi wprost w linii połączenia. Przydaje się do szybkiego testu połączenia z nowym serwerem, zanim jeszcze dodamy dla niego alias.

Przykład:

```
$ sqlplus app_user/********@ora01:1521/wmstest

SQL*Plus: Release 19.0.0.0.0 - Production on Thu Sep 25 08:22:47 2026

Connected to:
Oracle Database 19c Standard Edition 2 Release 19.22.0.0.0

APP_USER@wmstest: SQL>
```

sqlplus = SQL*Plus: klient bazy Oracle

TYLKO ODCZYT

Sprawdzenie stanu listenera: lsnrctl status

```
lsnrctl status
```

Przykład:

```
LSNRCTL for Linux: Version 19.0.0.0.0 - Production on 25-SEP-2026 08:30:02

Copyright (c) 1991, 2024, Oracle.  All rights reserved.

Connecting to (ADDRESS=(PROTOCOL=tcp)(HOST=ora01)(PORT=1521))
STATUS of the LISTENER
------------------------
Alias                     LISTENER
Version                   TNSLSNR for Linux: Version 19.0.0.0.0 - Production
Start Date                25-SEP-2026 06:00:14
Uptime                    0 days 2 hr. 29 min. 48 sec
Trace Level               off
Security                  ON: Local OS Authentication
SNMP                      OFF
Listener Parameter File   /oracle/db/19300/network/admin/listener.ora
Listener Log File         /oracle/db/diag/tnslsnr/ora01/listener/alert/log.xml
Listening Endpoints Summary...
  (DESCRIPTION=(ADDRESS=(PROTOCOL=tcp)(HOST=ora01)(PORT=1521)))
Services Summary...
Service "WMSPRD" has 1 instance(s).
  Instance "WMSPRD", status READY, has 1 handler(s) for this service.
The command completed successfully
```

Wiersz `Instance "WMSPRD", status READY` jest tym, co faktycznie sprawdzamy — `READY` oznacza, że listener widzi zarejestrowaną instancję i potrafi kierować do niej połączenia.

lsnrctl = listener control: sterowanie listenerem Oracle

TYLKO ODCZYT

Sprawdzenie usług zarejestrowanych w listenerze: lsnrctl services

```
lsnrctl services
```

Wersja bardziej szczegółowa od `status` — pokazuje też liczbę aktywnych handlerów i typ każdej usługi, przydatna, gdy `status` mówi, że wszystko jest `READY`, ale połączenia mimo to się nie udają.

Przykład:

```
Services Summary...
Service "WMSPRD" has 1 instance(s).
  Instance "WMSPRD", status READY, has 1 handler(s) for this service...
    Handler(s):
      "DEDICATED" established:1247 refused:0 state:ready
         LOCAL SERVER
The command completed successfully
```

lsnrctl = listener control: sterowanie listenerem Oracle

TYLKO ODCZYT

Uruchomienie listenera: lsnrctl start

```
lsnrctl start
```

Przykład:

```
LSNRCTL for Linux: Version 19.0.0.0.0 - Production on 25-SEP-2026 08:00:01

Copyright (c) 1991, 2024, Oracle.  All rights reserved.

Starting /oracle/db/19300/bin/tnslsnr: please wait...

TNSLSNR for Linux: Version 19.0.0.0.0 - Production
System parameter file is /oracle/db/19300/network/admin/listener.ora
Log messages written to /oracle/db/diag/tnslsnr/ora01/listener/alert/log.xml
Listening on: (DESCRIPTION=(ADDRESS=(PROTOCOL=tcp)(HOST=ora01)(PORT=1521)))

Connecting to (ADDRESS=(PROTOCOL=tcp)(HOST=ora01)(PORT=1521))
STATUS of the LISTENER
------------------------
Alias                     LISTENER
...
The command completed successfully
```

Samo sprawdzenie stanu poleceniem `lsnrctl status` bywa wystarczające, żeby listener wystartował, jeśli akurat był skonfigurowany na start automatyczny — ale nie jest to

zachowanie, na którym warto polegać świadomie; do uruchamiania służy `lsnrctl start`.

lsnrctl = listener control: sterowanie listenerem Oracle

TYLKO ODCZYT

Zatrzymanie listenera: lsnrctl stop

```
lsnrctl stop
```

Przykład:

```
LSNRCTL for Linux: Version 19.0.0.0.0 - Production on 25-SEP-2026 08:35:40

Copyright (c) 1991, 2024, Oracle.  All rights reserved.

Connecting to (ADDRESS=(PROTOCOL=tcp)(HOST=ora01)(PORT=1521))
The command completed successfully
```

Zatrzymanie listenera nie przerywa już nawiązanych sesji — blokuje wyłącznie nawiązywanie nowych połączeń sieciowych. Sesje lokalne na serwerze też nie ucierpią, bo te i tak nie przechodzą przez listener.

lsnrctl = listener control: sterowanie listenerem Oracle

Z życia wzięte

Z ŻYCIA WZIĘTE

Tomek loguje się na ora01 i odpala `sqlplus / as sysdba`, ale trafia na instancję WMSTEST zamiast WMSPRD, bo zapomniał wcześniej wykonać `. oraenv`. Marek tymczasem zgłasza, że aplikacja WMS w Swarzędzu zwraca błąd ORA-12154, bo ktoś zmienił `TNS_ADMIN` na serwerze aplikacyjnym i alias WMSPRD zniknął z widoku. Po poprawieniu zmiennej i ponownym uruchomieniu `lsnrctl status` połączenia wracają.

Zmienne środowiskowe Oracle

Pojęcie: ORACLE_HOME. Katalog instalacji binarek silnika — stamtąd `sqlplus`, `rman`, `lsnrctl` biorą się fizycznie. Jeden serwer potrafi mieć kilka `ORACLE_HOME` obok siebie, gdy trzymamy równolegle kilka wersji silnika.

Pojęcie: ORACLE_SID. Identyfikator instancji, do której łączy się `sqlplus / as sysdba` bez podawania danych połączenia. To jest zmienna, którą zmieniamy najczęściej — cała procedura przełączania się między bazami, opisana niżej, sprowadza się do zmiany tej jednej wartości.

Pojęcie: ORACLE_BASE. Katalog nadrzędny całej instalacji Oracle na serwerze — `ORACLE_HOME` jest zwykle jego podkatalogiem. Narzędzia instalacyjne i diagnostyczne Oracle (`adrci`, katalog `diag`) domyślnie piszą właśnie pod `ORACLE_BASE`.

Pozostałe zmienne środowiskowe mają węższe, ale równie istotne zastosowania:

Zmienna	Do czego służy
`PATH`	musi zawierać `$ORACLE_HOME/bin`, inaczej `sqlplus`/`rman`/ `lsnrctl` nie zostaną znalezione
`LD_LIBRARY_PATH`	ścieżka do bibliotek współdzielonych silnika; brakująca biblioteka kończy się błędem `error while loading shared libraries`
`NLS_LANG`	ustawia język komunikatów, kodowanie znaków i format daty/liczb sesji klienckiej
`TNS_ADMIN`	katalog, w którym `sqlplus` szuka plików `tnsnames.ora`, `sqlnet.ora` i `listener.ora`, jeśli nie leżą w domyślnym `$ORACLE_HOME/network/admin`

Plik oratab i skrypt oraenv

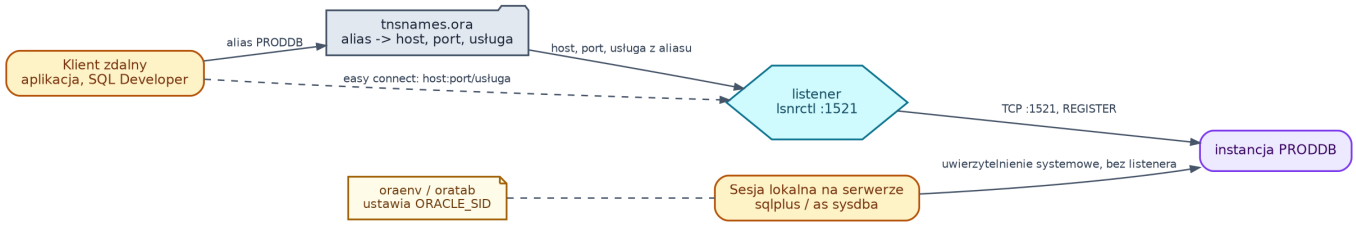
Pojęcie: oratab. Plik `/etc/oratab` (czasem `/var/opt/oracle/oratab`) to rejestr wszystkich instancji zainstalowanych na serwerze — jeden wiersz na instancję, w formacie `NazwaSID:SciezkaDoORACLE_HOME:Y_lub_N`. Trzecia kolumna mówi skryptom startowym, czy dana instancja ma się uruchamiać automatycznie razem z systemem.

Pojęcie: oraenv. Skrypt dostarczany razem z instalacją Oracle, który czyta `oratab`, pyta o `ORACLE_SID` i na tej podstawie ustawia `ORACLE_HOME`, `ORACLE_BASE` i `PATH` w bieżącej sesji

powłoki. Uruchamiamy go poleceniem `. oraenv` (z kropką na początku — inaczej zmienne wylądują w podpowłoce i znikną, gdy `oraenv` się zakończy).

Pliki konfiguracyjne TNS

Pojęcie: TNS (Transparent Network Substrate). Warstwa sieciowa Oracle, która tłumaczy nazwę usługi bazodanowej na adres, port i identyfikator serwisu potrzebny do faktycznego połączenia sieciowego. Trzy pliki konfigurują tę warstwę, każdy po innej stronie połączenia:



Sesja lokalna loguje się do instancji bez listenera, a klient zdalny dochodzi do niej przez alias z tnsnames.ora albo easy connect i listener nasłuchujący na porcie 1521 (kliknięcie otwiera pełny rozmiar)

Pojęcie: tnsnames.ora. Plik po stronie klienta — słownik aliasów. Definiuje, jaki alias (np. `WMSPRD`) odpowiada jakiemu adresowi, portowi i nazwie usługi na serwerze.

Pojęcie: listener.ora. Plik po stronie serwera — konfiguracja procesu nasłuchującego (`lsnrctl`): na jakim porcie i interfejsie nasłuchuje, jakie usługi (bazy) obsługuje.

Pojęcie: sqlnet.ora. Plik konfiguracji ogólnych parametrów sieciowych Oracle Net — metody uwierzytelniania, szyfrowanie transmisji, kolejność przeszukiwania metod rozwiązywania nazw (`tnsnames.ora` kontra katalogowa usługa nazw).

Wszystkie trzy pliki leżą domyślnie w `$ORACLE_HOME/network/admin`, chyba że zmienna `TNS_ADMIN` wskazuje inną lokalizację — typowo wspólny katalog konfiguracji, gdy kilka `ORACLE_HOME` ma współdzielić te same wpisy `tnsnames.ora`.

Listener — TNS LISTENER

> **Pojęcie: listener.** Proces nasłuchujący na porcie sieciowym (domyślnie 1521), który przyjmuje przychodzące połączenia i przekazuje je do właściwej instancji. Sesja lokalna `sqlplus / as sysdba` nie potrzebuje działającego listenera — problem z listenerem objawia się dopiero przy próbie połączenia „z zewnątrz": z aplikacji, z SQL Developera, z innego serwera.

Częste błędy

Zmiana ORACLE_SID bez uruchomienia oraenv przez kropkę. `bash oraenv` albo `sh oraenv` odpalają skrypt w podprocesie — zmienne środowiskowe ustawią się tam i znikną, gdy skrypt się skończy, a macierzysta powłoka zostanie z poprzednim `ORACLE_SID`. Uruchamiamy zawsze `. oraenv` albo `source oraenv`.

ORA-12154: TNS:could not resolve the connect identifier specified. Alias podany po `@` nie istnieje w żadnym pliku `tnsnames.ora` widocznym z bieżącego `TNS_ADMIN` — literówka w nazwie aliasu albo zła zmienna `TNS_ADMIN` wskazująca nie ten katalog, co trzeba.

ORA-12541: TNS:no listener. Adres i port się zgadzają, ale po drugiej stronie nikt nie nasłuchuje — najpierw `lsnrctl status` na serwerze docelowym, dopiero potem szukanie błędu po stronie klienta.

Powiązane: Architektura Oracle: instancja, baza, SGA, PGA, procesy tła · SQL*Plus: klient i jego konfiguracja · Rozpoznanie instancji: wersja, parametry, komponenty · Łącza bazodanowe (DB links)

Źródła: notatki własne, dokumentacja Oracle Net Services 19c

SQL*Plus: klient i jego konfiguracja

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

*Domyślne SQL*Plus obcina wynik do 80 znaków szerokości i nie pamięta poprzednich komend po strzałce w górę — obie rzeczy naprawiamy raz, w plikach konfiguracyjnych, i nie wracamy do tematu.*

Skróty użyte na tej stronie (1)

Skrót	Co znaczy
SQL	Structured Query Language: język zapytań do relacyjnych baz danych

Fiszka

ZMIENIA SYSTEM

Ustawienie szerokości linii: SET LINESIZE

```
SET LINESIZE 200
```

Liczba znaków, po której SQL*Plus łamie linię wyniku. Wartość 80 (domyślna w wielu instalacjach) wystarcza na dwie, trzy kolumny — 200 jest bezpiecznym punktem wyjścia dla większości zapytań administracyjnych.

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

ZMIENIA SYSTEM

Ustawienie liczby wierszy na stronę: SET PAGESIZE

```
SET PAGESIZE 50
```

Po ilu wierszach wyniku SQL*Plus powtarza nagłówki kolumn. Wartość `0` wyłącza nagłówki i podziały na strony całkowicie — przydatne przy generowaniu czystych danych do dalszego przetwarzania w pliku ze `SPOOL`.

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

TYLKO ODCZYT

Ustawienie formatu kolumny: COLUMN FORMAT

```
COLUMN username FORMAT A20
```

Wymuszamy szerokość konkretnej kolumny tekstowej na 20 znaków — bez tego SQL*Plus dobiera szerokość kolumny do najdłuższej wartości w danym typie danych zdefiniowanym w słowniku, co dla `VARCHAR2(4000)` oznacza kolumnę szeroką na całą linię. Format `A20` dotyczy tekstu, `999999` dotyczy liczb i ustala maksymalną liczbę cyfr.

Przykład:

```
SQL> COLUMN username FORMAT A20
SQL> SELECT username, account_status FROM dba_users WHERE ROWNUM <= 3;

USERNAME             ACCOUNT_STATUS
-------------------- --------------------------------
SYS                  OPEN
SYSTEM               OPEN
APP_USER             OPEN
```

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

ZMIENIA SYSTEM

Obcięcie białych znaków na końcu linii: SET TRIMSPOOL

```
SET TRIMSPOOL ON
```

Bez tego ustawienia każda linia zapisana przez `SPOOL` do pliku jest dopełniana spacjami do pełnej szerokości `LINESIZE` — `TRIMSPOOL ON` obcina te spacje, więc plik wynikowy nie puchnie bez potrzeby i lepiej się przetwarza narzędziami tekstowymi typu `grep` czy `awk`.

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

Wyłączenie stopki z liczbą wierszy: SET FEEDBACK

```
SET FEEDBACK OFF
```

Domyślnie SQL*Plus dopisuje na końcu wyniku linię w rodzaju `12 rows selected.` — przy generowaniu pliku wynikowego do dalszego przetwarzania ta linia jest szumem, który trzeba by potem odfiltrować. `SET FEEDBACK OFF` ją usuwa; `SET FEEDBACK ON` (domyślnie) przywraca.

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

ZMIENIA SYSTEM

Treść pliku glogin.sql do wklejenia

```
-- NAME
--   glogin.sql
--
-- DESCRIPTION
--   SQL*Plus global login "site profile" file
--   This script is automatically run
--   when a user starts SQL*Plus, or uses the SQL*Plus CONNECT command.
--   Add any SQL*Plus commands here that are to be executed when a
--   user starts SQL*Plus.
--
-- USAGE
--   /oracle/db/19300/sqlplus/admin/glogin.sql
--
-- Copyright (c) 1988, 2005, Oracle.  All Rights Reserved.
```

```
-- To set the number of characters to display on each report line to 200.
SET LINESIZE 200

-- To set the number of lines to display in a report page to 50.
SET PAGESIZE 50

-- SET the SQLPROMPT to include the _USER, _CONNECT_IDENTIFIER
-- and _DATE variables.
SET SQLPROMPT "_USER'@'_CONNECT_IDENTIFIER': SQL'> "

-- To set the format of columns
COLUMN username FORMAT A20
COLUMN status FORMAT A10
COLUMN account_status FORMAT A15
COLUMN grantee FORMAT A15
COLUMN member FORMAT A60
COLUMN file_name FOR A60
COLUMN value FORMAT A60
COLUMN next_change# FORMAT FM99999999999999999999
```

Linia `SET SQLPROMPT` jest tym, co zamienia domyślny, bezosobowy prompt `SQL>` na `SYS@WMSPRD: SQL>` — informację, jakim użytkownikiem i na jakiej instancji właśnie pracujemy, widoczną bez żadnego dodatkowego zapytania. Przy pracy na kilku instancjach naraz to jest jedno z najbardziej opłacalnych ustawień w całym pliku — trudno pomylić bazę produkcyjną z testową, gdy prompt mówi to wprost przy każdej linii.

Przykład promptu po wczytaniu tego glogin.sql:

```
SYS@WMSPRD: SQL>
```

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

□□ **TYLKO ODCZYT**

Dodanie aliasów z rlwrap do profilu konta oracle

```
alias sqlplus="rlwrap sqlplus"
alias rman="rlwrap rman"
```

Wpisy dodajemy na końcu `~/.bash_profile` konta `oracle` — od tego momentu każde uruchomienie `sqlplus` albo `rman` w interaktywnej powłoce przechodzi przez `rlwrap`, a strzałka w górę przywołuje poprzednio wpisane komendy SQL.

Przykład:

```
oracle@ora01:~$ sqlplus / as sysdba
SQL> SELECT sysdate FROM dual;
[strzałka w górę]
SQL> SELECT sysdate FROM dual;
```

SQL = Structured Query Language: język zapytań do relacyjnych baz danych · **alias** = nadaj poleceniu własny, krótszy skrót na czas sesji powłoki

TYLKO ODCZYT

Alias logowania jednym poleceniem: ssqlplus

```
alias ssqlplus='rlwrap sqlplus / as sysdba'
```

Skracamy najczęściej wpisywaną komendę logowania do jednego słowa — po zmianie `ORACLE_SID` przez `oraenv` samo `ssqlplus` loguje nas jako sysdba na bieżącą instancję, z historią komend włączoną.

alias = nadaj poleceniu własny, krótszy skrót na czas sesji powłoki

TYLKO ODCZYT

Opisanie struktury obiektu: DESCRIBE

```
DESCRIBE dba_users
```

Wypisuje listę kolumn widoku albo tabeli razem z typem danych i informacją, czy kolumna dopuszcza `NULL` — szybszy sposób sprawdzenia struktury niż zapytanie do `DBA_TAB_COLUMNS`.

Przykład:

```
SQL> DESCRIBE dba_users
 Name                          Null?    Type
 ----------------------------- -------- ----------------------------
 USERNAME                      NOT NULL VARCHAR2(128)
```

```
USER_ID                       NOT NULL NUMBER
PASSWORD                               VARCHAR2(4000)
ACCOUNT_STATUS                NOT NULL VARCHAR2(32)
EXPIRY_DATE                            DATE
DEFAULT_TABLESPACE            NOT NULL VARCHAR2(30)
CREATED                        NOT NULL DATE
```

TYLKO ODCZYT

Sprawdzenie wartości parametru instancji: SHOW PARAMETER

```
SHOW PARAMETER processes
```

Skrót do przeszukania `v$parameter` po fragmencie nazwy parametru — bez pisania `SELECT name, value FROM v$parameter WHERE name LIKE '%...%'` za każdym razem. Więcej o samych parametrach na stronie Rozpoznanie instancji: wersja, parametry, komponenty.

Przykład:

```
SQL> SHOW PARAMETER processes

NAME                                 TYPE        VALUE
------------------------------------ ----------- ------------------------------
aq_tm_processes                      integer     1
processes                            integer     2000
```

show = FortiOS: wypisuje konfigurację, w podstawowej postaci tylko ustawienia różniące się od domyślnych, a z wariantem full-configuration także wartości domyślne

TYLKO ODCZYT

Sprawdzenie bieżącego użytkownika sesji: SHOW USER

```
SHOW USER
```

Przykład:

```
SQL> SHOW USER
USER is "SYS"
```

□□ **show** = FortiOS: wypisuje konfigurację, w podstawowej postaci tylko ustawienia różniące się od domyślnych, a z wariantem full-configuration także wartości domyślne

□□ TYLKO ODCZYT

Sprawdzenie rozmiaru SGA: SHOW SGA

```
SHOW SGA
```

Przykład:

```
SQL> SHOW SGA

Total System Global Area 7180646608 bytes
Fixed Size                  8956112 bytes
Variable Size            1191182336 bytes
Database Buffers         5972688896 bytes
Redo Buffers                7819264 bytes
```

□□ **SGA** = System Global Area: wspólna pamięć instancji Oracle · **show** = FortiOS: wypisuje konfigurację, w podstawowej postaci tylko ustawienia różniące się od domyślnych, a z wariantem full-configuration także wartości domyślne

□□ TYLKO ODCZYT

Zapis wyniku sesji do pliku: SPOOL

```
SPOOL SciezkaDoPliku
```

Od tej komendy wszystko, co pojawia się na ekranie, jest równolegle zapisywane do wskazanego pliku — aż do wykonania `SPOOL OFF`, które zamyka plik. Standardowe użycie do zbierania wyniku audytu albo długiego raportu bez ręcznego kopiowania z terminala.

Przykład:

```
SQL> SPOOL /tmp/audyt_userow.txt
SQL> SELECT username, account_status FROM dba_users;
...
SQL> SPOOL OFF
```

□□ TYLKO ODCZYT

Wykonanie skryptu z pliku: @

```
@SciezkaDoPliku
```

Uruchamia plik `.sql` tak, jakbyśmy wklejali jego zawartość linia po linii do bieżącej sesji. Podwójny znak `@@` robi to samo, ale szuka pliku w tym samym katalogu, co skrypt wywołujący — przydatne, gdy jeden skrypt woła kolejne.

Przykład:

```
SQL> @/oracle/scripts/sprawdz_tablespace.sql
```

□□ ZMIENIA SYSTEM

Włączenie wypisywania z DBMS_OUTPUT: SET SERVEROUTPUT ON

```
SET SERVEROUTPUT ON
```

Bez tego ustawienia wywołania `DBMS_OUTPUT.PUT_LINE` wewnątrz bloków PL/SQL wykonują się bez błędu, ale nic nie pojawia się na ekranie — bufor wypełnia się i przepada. `SET SERVEROUTPUT ON` włącza jego wyświetlanie po zakończeniu bloku.

Przykład:

```
SQL> SET SERVEROUTPUT ON
SQL> BEGIN
  2    DBMS_OUTPUT.PUT_LINE('Test');
  3  END;
  4  /
Test


PL/SQL procedure successfully completed.
```

□□ **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

Zakończenie sesji SQL*Plus: EXIT

```
EXIT
```

Zamyka bieżącą sesję i zwraca sterowanie do powłoki systemowej. `QUIT` robi dokładnie to samo — obie komendy są równoważne.

SQL = Structured Query Language: język zapytań do relacyjnych baz danych · **exit** = zakończ bieżącą powłokę albo skrypt z podanym kodem wyjścia

TYLKO ODCZYT

Wszystkie omówione komendy SET

Komenda	Co robi
`SET LINESIZE n`	szerokość linii wyniku w znakach
`SET PAGESIZE n`	liczba wierszy między powtórzeniami nagłówka
`SET TRIMSPOOL ON`	obcina spacje na końcu linii zapisywanej przez SPOOL
`SET FEEDBACK OFF`	wyłącza stopkę „N rows selected."
`SET SERVEROUTPUT ON`	włącza wypisywanie DBMS_OUTPUT.PUT_LINE
`SET SQLPROMPT "..."`	zmienia treść znaku zachęty

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

Z życia wzięte

Z ŻYCIA WZIĘTE

O 8:15 Tomek uruchamia długi skrypt diagnostyczny na WMSPRD i wynik łamie się w połowie każdej linii, bo nikt nie ustawił `SET LINESIZE` na nowym koncie systemowym. Marek w tym samym czasie prosi o listę zablokowanych kont, a brak `SET FEEDBACK OFF` w skrypcie eksportu do pliku dokleja do wyniku zbędną stopkę z liczbą wierszy. Po dodaniu tych ustawień do `glogin.sql` każdy, kto loguje się na ora01, dostaje czytelny prompt od razu.

Formatowanie wyników zapytań

> **Pojęcie: SQL*Plus.** Klient tekstowy Oracle, uruchamiany komendą `sqlplus` — nie jest osobną aplikacją instalowaną dodatkowo, tylko częścią każdej instalacji silnika, dostępną od razu na serwerze bazodanowym. To właśnie w nim wykonujemy większość komend z tej książki.

Domyślne ustawienia SQL*Plus są dobrane pod terminal sprzed lat — wynik zapytania z kilkunastoma kolumnami łamie się w połowie linii i staje się nieczytelny, dopóki nie poprawimy kilku parametrów sesji.

Plik glogin.sql

> **Pojęcie: glogin.sql.** Globalny skrypt logowania SQL*Plus, uruchamiany automatycznie przy starcie każdej sesji, niezależnie od tego, jakim użytkownikiem i do jakiej bazy się logujemy. Leży w `$ORACLE_HOME/sqlplus/admin/glogin.sql` — jeden plik na całą instalację, więc ustawienia w nim obowiązują każdego, kto loguje się z tego `ORACLE_HOME`.

Plik login.sql

> **Pojęcie: login.sql.** Odpowiednik `glogin.sql`, ale lokalny — SQL*Plus szuka go w katalogu bieżącym oraz w katalogach wskazanych zmienną `SQLPATH`, i uruchamia go po `glogin.sql`, jeśli go znajdzie. Ustawienia z `login.sql` nadpisują te z `glogin.sql`. Trzymamy w nim rzeczy specyficzne dla jednego katalogu roboczego albo jednego projektu — a to, co ma obowiązywać wszędzie, zostaje w `glogin.sql`.

Historia komend: rlwrap

> **Pojęcie: rlwrap.** Nakładka na dowolne polecenie tekstowe, która dodaje mu obsługę historii wpisywanych linii przez strzałkę w górę oraz edycję linii poleceń — czegoś, czego natywne `sqlplus` i `rman` nie mają wbudowanego. Pakiet instalujemy z repozytorium systemowego (`dnf install rlwrap` / `apt install rlwrap`), a potem owijamy nim komendy w profilu powłoki konta `oracle`.

Częste błędy

Ustawienia w login.sql zamiast glogin.sql, przy oczekiwaniu, że zadziałają wszędzie. `login.sql` jest czytany tylko wtedy, gdy leży w katalogu, z którego uruchamiamy `sqlplus`, albo w ścieżce `SQLPATH` — z innego katalogu roboczego SQL*Plus go nie znajdzie, a ustawienia po prostu nie wejdą w życie.

Brak SET SERVEROUTPUT ON i wrażenie, że blok PL/SQL „nic nie zrobił". Blok wykonał się poprawnie, tylko jego wypisania nigdy nie trafiły na ekran — najpierw sprawdzamy to ustawienie, zanim zaczniemy debugować sam kod.

`rlwrap` **niezainstalowany na serwerze, a alias już dodany do profilu.** Powłoka zwróci `rlwrap: command not found` przy każdym uruchomieniu `sqlplus` — instalujemy pakiet przed dodaniem aliasów, nie po.

Powiązane: Zmienne środowiskowe, TNS i logowanie do instancji · Rozpoznanie instancji: wersja, parametry, komponenty · Architektura Oracle: instancja, baza, SGA, PGA, procesy tła

Źródła: notatki własne, dokumentacja SQL*Plus User's Guide and Reference 19c