Użytkownicy i uprawnienia

Role, uprawnienia, konta monitorujące i łącza bazodanowe.

- Użytkownicy, role i uprawnienia w Oracle
- Konto monitorujące dla Zabbixa
- Łącza bazodanowe (DB links)

Użytkownicy, role i uprawnienia w Oracle

Poziom: podstawy **Dotyczy: Oracle Database 19c** **Czas: ~25 min**

Trzy widoki słownikowe — `DBA_TAB_PRIVS`, `DBA_ROLE_PRIVS`, `DBA_SYS_PRIVS` — odpowiadają na dziewięćdziesiąt procent pytań o to, kto co może zrobić w bazie. Reszta tej strony to warianty i przypadki szczególne wokół tych trzech widoków.

Skróty użyte na tej stronie (5)

Skrót	Co znaczy
ACL	Access Control List: lista, kto ma jakie prawa do pliku lub zasobu
DBA	Database Administrator: administrator baz danych
JDWP	Java Debug Wire Protocol: protokół sieciowy, którym debugger, na przykład w SQL Developerze, podłącza się do sesji PL/SQL
SQL	Structured Query Language: język zapytań do relacyjnych baz danych
WMS	Warehouse Management System: system zarządzania magazynem

Fiszka

TYLKO ODCZYT

Sprawdzenie uprawnień obiektowych danego użytkownika: DBA_TAB_PRIVS

```
SELECT * FROM dba_tab_privs WHERE grantee = 'NazwaUsera';
```

Przykład:

```
GRANTEE              OWNER           TABLE_NAME           GRANTOR
PRIVILEGE  GRANTABLE HIERARCHY
-------------------- --------------- -------------------- --------------------
---------- --------- ---------
APP_USER              APP_OWNER       ZAMOWIENIA           APP_OWNER
SELECT     NO        NO
APP_USER              APP_OWNER       ZAMOWIENIA           APP_OWNER
INSERT     NO        NO
```

TYLKO ODCZYT

Sprawdzenie ról nadanych danemu użytkownikowi: DBA_ROLE_PRIVS

```
SELECT * FROM dba_role_privs WHERE grantee = 'NazwaUsera';
```

Przykład:

```
GRANTEE              GRANTED_ROLE                   ADMIN_OPTION DEFAULT_ROLE
-------------------- ------------------------------ ------------ ------------
APP_USER              CONNECT                         NO           YES
APP_USER              RESOURCE                        NO           YES
```

TYLKO ODCZYT

Sprawdzenie uprawnień systemowych danego użytkownika: DBA_SYS_PRIVS

```
SELECT * FROM dba_sys_privs WHERE grantee = 'NazwaUsera';
```

Przykład:

```
GRANTEE              PRIVILEGE                                ADMIN_OPTION
-------------------- ---------------------------------------- ------------
APP_USER              CREATE SESSION                           NO
APP_USER              CREATE TABLE                             NO
```

TYLKO ODCZYT

Sprawdzenie wszystkich uprawnień użytkownika, łącznie z odziedziczonymi przez role

```
WITH all_roles_for_user AS (
  SELECT DISTINCT CONNECT_BY_ROOT grantee AS granted_user, granted_role
  FROM dba_role_privs
  WHERE grantee IN (SELECT username FROM dba_users)
  CONNECT BY grantee = PRIOR granted_role
),
all_user_privs AS (
  -- Uprawnienia obiektowe nadane bezpośrednio użytkownikowi
  SELECT privilege, owner AS obj_owner, table_name AS obj_name,
         grantee AS username, grantee AS grant_target,
         grantable AS admin_or_grant_opt, hierarchy AS hierarchy_opt
  FROM dba_tab_privs
  UNION ALL
  -- Uprawnienia obiektowe nadane przez rolę
  SELECT privilege, owner AS obj_owner, table_name AS obj_name,
         r.granted_user AS username, dba_tab_privs.grantee AS grant_target,
         grantable AS admin_or_grant_opt, hierarchy AS hierarchy_opt
  FROM dba_tab_privs
  JOIN all_roles_for_user r ON r.granted_role = dba_tab_privs.grantee
  UNION ALL
  -- Uprawnienia systemowe nadane bezpośrednio użytkownikowi
  SELECT privilege, NULL, NULL, grantee AS username, grantee AS grant_target,
         admin_option, NULL
  FROM dba_sys_privs
  UNION ALL
  -- Uprawnienia systemowe nadane przez rolę
  SELECT privilege, NULL, NULL, r.granted_user AS username, dba_sys_privs.grantee AS
grant_target,
         admin_option, NULL
  FROM dba_sys_privs
  JOIN all_roles_for_user r ON r.granted_role = dba_sys_privs.grantee
)
SELECT privilege, obj_owner, obj_name,
       REPLACE(grant_target, username, 'Direct to user') AS grant_source,
       MAX(admin_or_grant_opt) AS admin_or_grant_opt,
       MAX(hierarchy_opt) AS hierarchy_opt
```

```
FROM all_user_privs
WHERE username = 'NazwaUsera'
GROUP BY privilege, obj_owner, obj_name, REPLACE(grant_target, username, 'Direct to
user');
```

Zapytanie łączy cztery źródła naraz: uprawnienia obiektowe i systemowe nadane wprost, oraz te same dwa typy odziedziczone przez którąkolwiek z ról użytkownika, włącznie z rolami zagnieżdżonymi w innych rolach (`CONNECT BY` po `dba_role_privs`). Jest to jeden logiczny skrypt diagnostyczny, więc uruchamiamy go w całości, a nie po kawałku.

Przykład:

```
PRIVILEGE            OBJ_OWNER       OBJ_NAME             GRANT_SOURCE
ADMIN_OR_GRANT_OPT HIERARCHY_OPT
-------------------- --------------- -------------------- ---------------
------------------ -------------
CREATE SESSION                                            Direct to user   NO
SELECT               APP_OWNER       ZAMOWIENIA           Direct to user
NO                  NO
SELECT ANY DICTIONARY                                     SELECT_ANY_DICT  NO
```

join = połącz dwa pliki tekstowe po wspólnej kolumnie, jak złączenie tabel w bazie danych

TYLKO ODCZYT

Sprawdzenie kont z rolami systemowymi: v$pwfile_users

```
SELECT username, sysdba, sysoper, sysasm, sysbackup, sysdg, syskm FROM v$pwfile_users;
```

Przykład:

```
USERNAME             SYSDBA SYSOPER SYSASM SYSBACKUP SYSDG SYSKM
-------------------- ------ ------- ------ --------- ----- -----
SYS                  TRUE   TRUE    FALSE  FALSE     FALSE FALSE
APP_DBA               TRUE   FALSE   FALSE  FALSE     FALSE FALSE
```

TYLKO ODCZYT

Sprawdzenie daty wygasania kont

```
SELECT username, account_status, expiry_date FROM dba_users;
```

> **Pojęcie: profil hasła (password profile).** Zbiór ograniczeń dotyczących haseł — maksymalny czas życia, liczba nieudanych prób logowania przed zablokowaniem konta, wymuszona złożoność — przypisywany do użytkownika przy `CREATE USER ... PROFILE NazwaProfilu`. To profil, nie sam Oracle, decyduje, czy i kiedy `expiry_date` w ogóle zostanie ustawiona.

Przykład:

```
USERNAME             ACCOUNT_STATUS                 EXPIRY_DATE
-------------------- ------------------------------ --------------
APP_USER              OPEN                           25-MAR-27
REPORTS_USER          EXPIRED                        10-JUN-26
```

□□ TYLKO ODCZYT

Sprawdzenie tylko kont z wygasłym hasłem

```
SELECT username, account_status, expiry_date FROM dba_users WHERE account_status =
'EXPIRED';
```

Przykład:

```
USERNAME             ACCOUNT_STATUS                 EXPIRY_DATE
-------------------- ------------------------------ --------------
REPORTS_USER          EXPIRED                        10-JUN-26
```

□□ TYLKO ODCZYT

Pobranie definicji DDL użytkownika razem z haszem hasła

```
SELECT DBMS_METADATA.GET_DDL('USER', 'NazwaUsera') FROM dual;
```

Przykład:

```
DBMS_METADATA.GET_DDL('USER','NAZWAUSERA')
--------------------------------------------------------------------------------
```

```
CREATE USER "NAZWAUSERA" IDENTIFIED BY VALUES 'S:9F6B175C4B751AA871D440FFD00EBF35
40AA32D7A9E25BE8176099366032;36EB0E8BF249FE7B'
```

□□ **DDL** = Data Definition Language: polecenia SQL zmieniające strukturę (CREATE, ALTER, DROP)

□□ ZMIENIA SYSTEM

Ustawienie hasła z powrotem na dokładnie tę samą wartość

```
ALTER USER NazwaUsera IDENTIFIED BY VALUES
'S:9F6B175C4B751AA871D440FFD00EBF3540AA32D7A9E25BE8176099366032;36EB0E8BF249FE7B';
```

Klauzula `IDENTIFIED BY VALUES` przyjmuje bezpośrednio hasz zamiast hasła jawnego — to jest dokładnie ten sam hasz, jaki dostaliśmy z `DBMS_METADATA.GET_DDL` powyżej. Efekt końcowy to konto z identycznym hasłem jak przed operacją, mimo że w międzyczasie mogliśmy je odblokować, przenieść między bazami przez eksport/import albo zresetować przez pomyłkę.

Przykład:

```
SQL> ALTER USER NazwaUsera IDENTIFIED BY VALUES
'S:9F6B175C4B751AA871D440FFD00EBF3540AA32D7A9E25BE8176099366032;36EB0E8BF249FE7B';


User altered.
```

□□ ZMIENIA SYSTEM

Nadanie uprawnień do wszystkich obiektów danego właściciela

```
DECLARE
  v_sql VARCHAR2(1000);
BEGIN
  FOR priv IN (SELECT privilege, table_name, grantee
               FROM dba_tab_privs
               WHERE owner = 'NazwaWlasciciela') LOOP
    v_sql := 'GRANT ' || priv.privilege || ' ON NazwaWlasciciela.' || priv.table_name ||
' TO NazwaUsera';
    EXECUTE IMMEDIATE v_sql;
  END LOOP;
END;
```

```
/
```

Blok PL/SQL przegląda wszystkie uprawnienia obiektowe już nadane komukolwiek do obiektów danego właściciela (`dba_tab_privs WHERE owner = ...`) i powiela dokładnie ten sam zestaw uprawnień na nowego użytkownika. Praktyczne zastosowanie: nowy użytkownik ma dostać dokładnie takie same uprawnienia do schematu aplikacyjnego, jakie ma już inny, istniejący użytkownik.

Przykład:

```
SQL> DECLARE
  2    v_sql VARCHAR2(1000);
  3  BEGIN
  4    FOR priv IN (SELECT privilege, table_name, grantee
  5                 FROM dba_tab_privs
  6                 WHERE owner = 'APP_OWNER') LOOP
  7      v_sql := 'GRANT ' || priv.privilege || ' ON APP_OWNER.' || priv.table_name || '
TO APP_USER2';
  8      EXECUTE IMMEDIATE v_sql;
  9    END LOOP;
 10  END;
 11  /

PL/SQL procedure successfully completed.
```

declare = zadeklaruj zmienną powłoki z określonym typem, na przykład jako liczbę całkowitą albo tablicę · **execute** = FortiOS: polecenia, które coś wykonują na urządzeniu, na przykład execute backup config wysyła kopię konfiguracji, a execute ha manage otwiera sesję na drugiej zaporze w parze

ZMIENIA SYSTEM

Pętla nadająca jedno uprawnienie do wszystkich tabel użytkownika X dla użytkownika Y

```
BEGIN
  FOR t IN (SELECT table_name FROM dba_tables WHERE owner = 'NazwaUsera1') LOOP
    EXECUTE IMMEDIATE 'GRANT SELECT ON NazwaUsera1.' || t.table_name || ' TO
NazwaUsera2';
  END LOOP;
```

```
END;
/
```

W odróżnieniu od poprzedniego bloku, który powiela już istniejące uprawnienia, ta pętla nadaje jedno wybrane uprawnienie (tu `SELECT`) na wszystkie tabele danego schematu naraz, niezależnie od tego, czy ktokolwiek miał już do nich dostęp wcześniej — sięgamy po nią, gdy nowy użytkownik ma dostać czysty odczyt do całego schematu, a nie kopię cudzych uprawnień.

Przykład:

```
SQL> BEGIN
  2    FOR t IN (SELECT table_name FROM dba_tables WHERE owner = 'APP_OWNER') LOOP
  3      EXECUTE IMMEDIATE 'GRANT SELECT ON APP_OWNER.' || t.table_name || ' TO
REPORTS_USER';
  4    END LOOP;
  5  END;
  6  /

PL/SQL procedure successfully completed.
```

□□ **execute** = FortiOS: polecenia, które coś wykonują na urządzeniu, na przykład execute backup config wysyła kopię konfiguracji, a execute ha manage otwiera sesję na drugiej zaporze w parze

□□ **TYLKO ODCZYT**

Sprawdzenie uprawnień do directory

```
SELECT grantee, table_name AS directory_name, privilege
FROM dba_tab_privs
WHERE table_name = 'NazwaDirectory';
```

Przykład:

```
GRANTEE              DIRECTORY_NAME       PRIVILEGE
-------------------- -------------------- ----------
APP_USER              DATAPUMP_DIR          READ
APP_USER              DATAPUMP_DIR          WRITE
```

🔒 TYLKO ODCZYT

Zezwolenie w ACL na sesję debugowania w SQL Developer

```
BEGIN
  DBMS_NETWORK_ACL_ADMIN.APPEND_HOST_ACE(
    host          => '*',
    ace           => xs$ace_type(
                       privilege_list => xs$name_list('JDWP'),
                       principal_name => 'NazwaUsera',
                       principal_type => xs_acl.ptype_db)
  );
END;
/
```

Sesje debugowania kodu PL/SQL z poziomu SQL Developera łączą się do bazy przez protokół JDWP, którym od wersji 12c zarządza lista kontroli dostępu sieciowego (ACL), a nie już sam `GRANT`.

Przykład:

```
SQL> BEGIN
  2    DBMS_NETWORK_ACL_ADMIN.APPEND_HOST_ACE(
  3      host          => '*',
  4      ace           => xs$ace_type(
  5                         privilege_list => xs$name_list('JDWP'),
  6                         principal_name => 'APP_USER',
  7                         principal_type => xs_acl.ptype_db)
  8    );
  9  END;
 10  /

PL/SQL procedure successfully completed.
```

🔒 **ACL** = Access Control List: lista, kto ma jakie prawa do pliku lub zasobu · **SQL** = Structured Query Language: język zapytań do relacyjnych baz danych · **JDWP** = Java Debug Wire Protocol: protokół sieciowy, którym debugger, na przykład w SQL Developerze, podłącza się do sesji PL/SQL

🔒 ZMIENIA SYSTEM

Nadanie dwóch uprawnień klasycznych wymaganych obok wpisu w ACL

```
GRANT EXECUTE ON SYS.DBMS_DEBUG_JDWP TO NazwaUsera;
GRANT DEBUG CONNECT SESSION TO NazwaUsera;
```

Sama pozycja w ACL nie wystarcza — użytkownik potrzebuje jeszcze dwóch klasycznych uprawnień obok niej.

Przykład:

```
SQL> GRANT EXECUTE ON SYS.DBMS_DEBUG_JDWP TO APP_USER;

Grant succeeded.

SQL> GRANT DEBUG CONNECT SESSION TO APP_USER;

Grant succeeded.
```

ACL = Access Control List: lista, kto ma jakie prawa do pliku lub zasobu

TYLKO ODCZYT

Weryfikacja wpisu w ACL

```
SELECT principal, privilege FROM dba_network_acl_privileges WHERE principal =
'NazwaUsera';
```

Przykład:

```
PRINCIPAL            PRIVILEGE
-------------------- ----------
APP_USER              JDWP
```

ACL = Access Control List: lista, kto ma jakie prawa do pliku lub zasobu

Z życia wzięte

Z ŻYCIA WZIĘTE

Ola prosi o konto z dostępem tylko do odczytu danych WMS dla nowego raportu, na chwilę, do testów. Tomek sprawdza w `DBA_ROLE_PRIVS`, że podobne konto założone pół roku wcześniej wciąż ma rolę DBA, której nikt nie cofnął po zakończeniu testów. Zamiast kopiować stare uprawnienia, zakłada REPORTS_USER z samym SELECT na potrzebne tabele w APP_OWNER.

Konta i uprawnienia — podstawy

Pojęcie: użytkownik (user) w Oracle. W przeciwieństwie do PostgreSQL, gdzie rola i użytkownik to w gruncie rzeczy ten sam byt, Oracle rozróżnia konto (`CREATE USER`), które służy do logowania, i rolę (`CREATE ROLE`), która jest tylko nośnikiem uprawnień grupowanych razem. Konto może mieć przypisane role, ale samo logowanie zawsze odbywa się jako konkretny użytkownik.

Pojęcie: uprawnienie systemowe (system privilege). Uprawnienie dotyczące całej bazy albo szerokiej kategorii obiektów, nie jednego konkretnego — `CREATE SESSION`, `CREATE TABLE`, `SELECT ANY DICTIONARY`. Widoczne w `DBA_SYS_PRIVS`.

Pojęcie: uprawnienie obiektowe (object privilege). Uprawnienie dotyczące jednego konkretnego obiektu — `SELECT` na konkretnej tabeli, `EXECUTE` na konkretnej procedurze. Widoczne w `DBA_TAB_PRIVS`.

Pojęcie: rola (role). Zbiór uprawnień systemowych i obiektowych pod jedną nazwą, nadawany użytkownikowi jednym poleceniem `GRANT NazwaRoli TO NazwaUsera` zamiast wielu osobnych `GRANT`. Cztery role wbudowane pojawiają się praktycznie w każdej instalacji:

Rola	Co daje
`CONNECT`	prawo do zalogowania się (`CREATE SESSION`)
`RESOURCE`	prawo do tworzenia własnych obiektów (tabel, procedur, sekwencji)
`DBA`	pełne uprawnienia administracyjne w obrębie bazy/kontenera
`SELECT_ANY_DICTIONARY`	odczyt dowolnego widoku słownika danych (`DBA_*`), bez uprawnień do danych aplikacyjnych

Role systemowe (SYSDBA, SYSOPER)

> **Pojęcie: rola systemowa.** W odróżnieniu od zwykłych ról bazodanowych, role systemowe (`SYSDBA`, `SYSOPER`, `SYSBACKUP`, `SYSDG`, `SYSKM`, `SYSASM`) uprawniają do administrowania samą instancją — w tym do połączenia się z bazą, zanim jeszcze jest ona otwarta. Nadawane są przez plik haseł (`orapwd`), nie przez zwykły `GRANT` do schematu.

Reset hasła bez zmiany jego wartości

Czasem potrzebujemy odblokować konto albo przenieść je na inny serwer, nie znając samego hasła w postaci jawnej — i nie chcąc go zmieniać dla aplikacji, która się nim łączy. Oracle pozwala na to, bo `DBMS_METADATA` potrafi wyciągnąć hasz hasła wprost ze słownika.

Directories i ACL sieciowe

> **Pojęcie: directory (obiekt bazy).** Wskaźnik na katalog systemu plików serwera, widoczny dla bazy pod nazwą logiczną — wykorzystywany m.in. przez `DATAPUMP` do zapisu plików eksportu. Uprawnienia do directory są uprawnieniami obiektowymi jak każde inne, widocznymi w `DBA_TAB_PRIVS`.

Częste błędy

ORA-28000: the account is locked. Konto przekroczyło limit nieudanych prób logowania zdefiniowany w profilu haseł — odblokowujemy przez `ALTER USER NazwaUsera ACCOUNT UNLOCK`, ale samo odblokowanie nie zmienia hasła.

Reset hasła przez zwykły ALTER USER ... IDENTIFIED BY zamiast IDENTIFIED BY VALUES. Zwykłe `IDENTIFIED BY` ustawia nowe hasło jawne i bezpowrotnie nadpisuje stary hasz — jeśli celem jest zachowanie dokładnie tego samego hasła, jedyną drogą jest odzyskanie hasza przez `DBMS_METADATA.GET_DDL` i użycie `IDENTIFIED BY VALUES`.

Zapomnienie o EXECUTE ON DBMS_DEBUG_JDWP przy konfiguracji ACL. Sam wpis w ACL bez tego uprawnienia nie wystarcza do uruchomienia sesji debugowania — SQL Developer zwróci błąd połączenia z debuggerem, mylnie sugerujący problem sieciowy.

Powiązane: Konta, role i uprawnienia: GRANT, REVOKE, zasada najmniejszych uprawnień · Konto monitorujące dla Zabbixa · Architektura Oracle: instancja, baza, SGA, PGA, procesy tła

Źródła: notatki własne, dokumentacja Oracle Database Security Guide 19c

Konto monitorujące dla Zabbixa

□□ Typ: procedura **□□ Dotyczy: Oracle Database 19c, szablon Zabbix Oracle**

□ Czas: ~20 min

Efekt: konto `ZABBIX` (albo `C##ZABBIX` w bazie kontenerowej) z odczytem do widoków wydajnościowych i słownikowych, wystarczającym dla oficjalnego szablonu monitoringu Zabbixa, bez uprawnień do modyfikacji czegokolwiek w bazie.

□□ Skróty użyte na tej stronie (5)

Skrót	Co znaczy
CDB	Container Database: baza-kontener w Oracle 12c i nowszych, trzyma w sobie bazy PDB
ODBC	Open Database Connectivity: uniwersalny sposób łączenia się z różnymi bazami danych tym samym interfejsem
PDB	Pluggable Database: baza wpinana do kontenera CDB w Oracle
README	Read Me: plik tekstowy z opisem projektu, wykresu lub szablonu, czytany jako pierwszy
WMS	Warehouse Management System: system zarządzania magazynem

Zanim zaczniemy

> **□□ Z ŻYCIA WZIĘTE**
>
> Zanim zespół dorzuci ora01 do Zabbixa, Marek pyta, czy monitoring nie spowolni WMS w godzinach szczytu. Tomek zakłada konto ZABBIX z samym `SELECT_CATALOG_ROLE` i garścią uprawnień do widoków `v$`, bez żadnego prawa zapisu. Kiedy szablon zaczyna pobierać metryki co minutę, obciążenie instancji WMSPRD nie zmienia się zauważalnie.

Wymaganie	Jak sprawdzić
Dostęp jako sysdba do instancji docelowej	`sqlplus / as sysdba`
Ustalone hasło dla konta ZABBIX	wg polityki haseł ustalonej dla kont technicznych

Wymaganie	Jak sprawdzić
Wiedza, czy baza jest klasyczna czy kontenerowa (CDB)	`SELECT cdb FROM v$database;`

Uważaj: konto monitorujące dostaje wyłącznie uprawnienia `SELECT` i rolę `SELECT_CATALOG_ROLE` — nigdy `DBA` ani uprawnień do zapisu. Szablon Zabbixa tylko odczytuje metryki, więc szersze uprawnienia byłyby czystym ryzykiem bez żadnej korzyści.

Procedura ma dwa warianty w zależności od architektury instancji — klasyczny (baza niekontenerowa) i kontenerowy (CDB z użytkownikiem wspólnym `C##`). Rozróżnienie CDB/PDB opisuje strona Architektura Oracle: instancja, baza, SGA, PGA, procesy tła.

Przebieg — wariant klasyczny (baza niekontenerowa)

1. Utworzenie konta

```
CREATE USER zabbix IDENTIFIED BY "HasloZabbixa";
```

```
User created.
```

2. Nadanie prawa logowania

```
GRANT CONNECT, CREATE SESSION TO zabbix;
```

```
Grant succeeded.
```

3. Nadanie roli odczytu słownika danych

```
GRANT SELECT_CATALOG_ROLE TO zabbix;
```

> **Pojęcie: SELECT_CATALOG_ROLE.** Rola wbudowana dająca odczyt do praktycznie całego słownika danych (`DBA_*`) jedną komendą, zamiast nadawania `SELECT` na dziesiątki widoków z osobna. Szablon Zabbixa odpytuje spory zestaw widoków `DBA_*`, więc ta rola jest tu

wygodniejsza niż wyliczanie ich pojedynczo.

```
Grant succeeded.
```

4. Nadanie odczytu do widoków dynamicznych wydajności

Każdy `GRANT` uruchamiamy osobno — to nie jest sekwencja, w której kolejność ma znaczenie, ale gdy któryś widok nie istnieje w danej wersji silnika (rzadkie, ale zdarza się przy starszych patchach), pojedynczy błąd nie przerywa reszty.

```
GRANT SELECT ON v_$database TO zabbix;
GRANT SELECT ON v_$instance TO zabbix;
GRANT SELECT ON v_$session TO zabbix;
GRANT SELECT ON v_$process TO zabbix;
GRANT SELECT ON v_$sysmetric TO zabbix;
GRANT SELECT ON v_$sgastat TO zabbix;
GRANT SELECT ON v_$pgastat TO zabbix;
GRANT SELECT ON v_$system_parameter TO zabbix;
GRANT SELECT ON v_$osstat TO zabbix;
GRANT SELECT ON v_$log TO zabbix;
GRANT SELECT ON v_$datafile TO zabbix;
GRANT SELECT ON v_$archive_dest TO zabbix;
GRANT SELECT ON v_$recovery_file_dest TO zabbix;
GRANT SELECT ON v_$restore_point TO zabbix;
GRANT SELECT ON v_$active_session_history TO zabbix;
GRANT SELECT ON v_$asm_diskgroup TO zabbix;
GRANT SELECT ON sys.dba_data_files TO zabbix;
GRANT SELECT ON DBA_USERS TO zabbix;
GRANT SELECT ON DBA_TABLESPACES TO zabbix;
GRANT SELECT ON DBA_TABLESPACE_USAGE_METRICS TO zabbix;
```

```
Grant succeeded.
Grant succeeded.
...
Grant succeeded.
```

Przebieg — wariant kontenerowy (CDB, konto C##ZABBIX)

W architekturze CDB konto wspólne musi mieć nazwę z prefiksem `C##` — wymóg silnika opisany na stronie Architektura Oracle: instancja, baza, SGA, PGA, procesy tła. Zamiast widoków `DBA_*`, ograniczonych do bieżącego kontenera, nadajemy uprawnienia do ich odpowiedników `CDB_*`, widocznych ze wszystkich PDB naraz.

1. Utworzenie konta wspólnego

```
CREATE USER c##zabbix IDENTIFIED BY "HasloZabbixa" CONTAINER=ALL;
```

```
User created.
```

2. Nadanie prawa logowania

```
GRANT CONNECT, CREATE SESSION TO c##zabbix;
```

```
Grant succeeded.
```

3. Nadanie roli odczytu słownika i jej ograniczenie do bieżącego kontenera

```
GRANT SELECT_CATALOG_ROLE TO c##zabbix;
```

```
Grant succeeded.
```

```
ALTER USER c##zabbix SET CONTAINER_DATA=ALL CONTAINER=CURRENT;
```

`CONTAINER_DATA=ALL` pozwala kontu wspólnemu widzieć dane wszystkich kontenerów w widokach `CDB_*`, zamiast tylko tego, w którym akurat jest zalogowane.

```
User altered.
```

4. Nadanie odczytu do widoków dynamicznych i CDB_*

```
GRANT SELECT ON v_$database TO c##zabbix;
GRANT SELECT ON v_$instance TO c##zabbix;
GRANT SELECT ON v_$session TO c##zabbix;
GRANT SELECT ON v_$process TO c##zabbix;
GRANT SELECT ON v_$parameter TO c##zabbix;
GRANT SELECT ON v_$system_parameter TO c##zabbix;
GRANT SELECT ON v_$sysmetric TO c##zabbix;
GRANT SELECT ON v_$sgastat TO c##zabbix;
GRANT SELECT ON v_$pgastat TO c##zabbix;
GRANT SELECT ON v_$osstat TO c##zabbix;
GRANT SELECT ON v_$log TO c##zabbix;
GRANT SELECT ON v_$datafile TO c##zabbix;
GRANT SELECT ON v_$archive_dest TO c##zabbix;
GRANT SELECT ON v_$recovery_file_dest TO c##zabbix;
GRANT SELECT ON v_$active_session_history TO c##zabbix;
GRANT SELECT ON v_$asm_diskgroup TO c##zabbix;
GRANT SELECT ON v_$asm_diskgroup_stat TO c##zabbix;
GRANT SELECT ON v_$sort_segment TO c##zabbix;
GRANT SELECT ON v_$pdbs TO c##zabbix;
GRANT SELECT ON DBA_USERS TO c##zabbix;
GRANT SELECT ON CDB_TABLESPACES TO c##zabbix;
GRANT SELECT ON CDB_DATA_FILES TO c##zabbix;
GRANT SELECT ON CDB_TEMP_FILES TO c##zabbix;
GRANT SELECT ON CDB_FREE_SPACE TO c##zabbix;
```

```
Grant succeeded.
Grant succeeded.
...
Grant succeeded.
```

5. Wycofanie SELECT_CATALOG_ROLE po ustabilizowaniu monitoringu (opcjonalnie)

W jednym z wdrożeń, po przejściu na jawnie wyliczone uprawnienia do widoków `CDB_*`, szeroka rola `SELECT_CATALOG_ROLE` z kroku 3 przestała być potrzebna i została wycofana — zasada najmniejszych uprawnień z czasem, nie tylko na starcie:

```
REVOKE SELECT_CATALOG_ROLE FROM c##zabbix;
```

```
Revoke succeeded.
```

Ten krok jest opcjonalny i wykonujemy go dopiero po potwierdzeniu, że szablon Zabbixa faktycznie działa na jawnie nadanych uprawnieniach z kroku 4 — nie przed weryfikacją.

Funkcja pomocnicza wymagana przez szablon Zabbixa: punkty przywracania w CDB (opcjonalnie)

Widok `V$RESTORE_POINT` w architekturze CDB nie jest bezpośrednio widoczny w formie, jakiej oczekuje oficjalny szablon Zabbixa dla Oracle — dlatego szablon (zobacz źródło w sekcji Źródła) definiuje własny typ obiektowy i funkcję potokową (pipelined), która odczytuje `V$RESTORE_POINT` i zwraca jej wiersze w ujednoliconej formie, niezależnie od kontenera.

```
CREATE OR REPLACE TYPE zbx_mon_restore_point_row AS OBJECT (
  scn                          NUMBER,
  database_incarnation#        NUMBER,
  guarantee_flashback_database VARCHAR2(3),
  storage_size                 NUMBER,
  time                         TIMESTAMP(9),
  restore_point_time           TIMESTAMP(9),
  preserved                    VARCHAR2(3),
  name                         VARCHAR2(128),
  pdb_restore_point            VARCHAR2(3),
  clean_pdb_restore_point      VARCHAR2(3),
  pdb_incarnation#             NUMBER,
  replicated                   VARCHAR2(3),
  con_id                       NUMBER
);
/
```

```
CREATE OR REPLACE TYPE zbx_mon_restore_point_tab IS TABLE OF zbx_mon_restore_point_row;
/

CREATE OR REPLACE FUNCTION zbx_mon_restore_point RETURN zbx_mon_restore_point_tab PIPELINED AS
BEGIN
  FOR i IN (SELECT * FROM v$restore_point) LOOP
    PIPE ROW (zbx_mon_restore_point_row(
      i.scn, i.database_incarnation#, i.guarantee_flashback_database, i.storage_size,
      i.time, i.restore_point_time, i.preserved, i.name, i.pdb_restore_point,
      i.clean_pdb_restore_point, i.pdb_incarnation#, i.replicated, i.con_id));
  END LOOP;
  RETURN;
END;
/
```

```
Type created.
Type created.
Function created.
```

Instalujemy tę funkcję tylko wtedy, gdy konkretna wersja szablonu Zabbixa jej wymaga — sprawdzamy to w dokumentacji szablonu, nie zakładamy z góry.

Weryfikacja

Sprawdzenie, że konto się loguje i widzi widoki

```
CONNECT zabbix/HasloZabbixa@NazwaInstancji
SELECT instance_name, status FROM v$instance;
```

Przykład (wariant klasyczny):

```
SQL> CONNECT zabbix/********@WMSPRD
Connected.
SQL> SELECT instance_name, status FROM v$instance;
```

```
INSTANCE_NAME    STATUS
---------------- ------------
WMSPRD           OPEN
```

Sprawdzenie kompletu nadanych uprawnień

```
SELECT privilege, table_name FROM dba_tab_privs WHERE grantee = 'ZABBIX' ORDER BY table_name;
```

Przykład:

```
PRIVILEGE  TABLE_NAME
---------- ------------------------------
SELECT     V_$ARCHIVE_DEST
SELECT     V_$DATABASE
SELECT     V_$INSTANCE
SELECT     V_$SESSION
...
```

Sprawdzenie, że konto nie ma nic ponad odczyt

```
SELECT privilege FROM dba_sys_privs WHERE grantee = 'ZABBIX';
```

Przykład:

```
PRIVILEGE
----------------------------------------
CREATE SESSION
```

Jedyne uprawnienie systemowe, jakie konto monitorujące powinno mieć bezpośrednio, to `CREATE SESSION` — wszystko inne dochodzi przez `SELECT_CATALOG_ROLE` albo jawne `GRANT SELECT` na widoki wymienione wyżej.

Wycofanie zmian

```
DROP USER zabbix CASCADE;
```

albo, w wariancie kontenerowym:

```
DROP USER c##zabbix CASCADE;
```

`CASCADE` usuwa razem z kontem wszystkie obiekty, jakie zdążyło ono utworzyć — dla konta czysto monitorującego, bez własnych tabel, nie zmienia to nic poza samym kontem, ale trzymamy się tej klauzuli jako standardu, żeby procedura była identyczna niezależnie od tego, czy ktoś w międzyczasie dołożył kontu cokolwiek własnego.

```
User dropped.
```

Gdy coś nie zadziała

Objaw	Przyczyna	Naprawa
`ORA-01031: insufficient privileges` przy GRANT na v_$...	wykonanie GRANT zalogowanym jako zwykły DBA bez SYSDBA, na wersji z zaostrzonymi uprawnieniami do widoków v$	logujemy się `sqlplus / as sysdba`
`ORA-00942: table or view does not exist` przy CREATE USER z prefiksem C##	próba utworzenia konta z prefiksem C## na bazie, która nie jest CDB	sprawdzamy `SELECT cdb FROM v$database;` przed wyborem wariantu
Szablon Zabbixa zgłasza brak dostępu do konkretnej metryki mimo wykonanych GRANT-ów	widok dodany w nowszej wersji szablonu, którego nie ma na liście w kroku 4	sprawdzamy README aktualnej wersji szablonu i donadajemy brakujący `GRANT SELECT`
`ORA-01920: user name 'C##ZABBIX' conflicts with another user or role name`	konto już istnieje z poprzedniej próby instalacji	`DROP USER c##zabbix CASCADE;` i powtórzenie procedury od początku

Powiązane: Użytkownicy, role i uprawnienia · Architektura Oracle: instancja, baza, SGA, PGA, procesy tła · Co monitorujemy w bazie danych

Źródła: notatki własne, README szablonu Zabbix Oracle ODBC (git.zabbix.com, gałąź release/7.0)

Łącza bazodanowe (DB links)

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

Database link pozwala odpytać obcą bazę z poziomu zwykłego `SELECT`, tak jakby jej tabele leżały lokalnie — kosztem tego, że dane logowania do tej obcej bazy trzeba gdzieś trwale przechować.

Skróty użyte na tej stronie (3)

Skrót	Co znaczy
SQL	Structured Query Language: język zapytań do relacyjnych baz danych
TNS	Transparent Network Substrate: warstwa sieciowa Oracle, przez którą klient znajduje i łączy się z listenerem bazy
WMS	Warehouse Management System: system zarządzania magazynem

Fiszka

ZMIENIA SYSTEM

Utworzenie łącza prywatnego z podaniem hasła wprost

```
CREATE DATABASE LINK NazwaLinku
CONNECT TO NazwaUsera IDENTIFIED BY HasloUsera
USING '(DESCRIPTION=
          (ADDRESS=(PROTOCOL=TCP)(HOST=NazwaSerwera)(PORT=1521))
          (CONNECT_DATA=(SERVICE_NAME=NazwaInstancji))
        )';
```

Adres docelowy podajemy tu wprost, w opisowej składni Oracle Net, tej samej, jaką widzielibyśmy w `tnsnames.ora` — link nie musi korzystać z aliasu zdefiniowanego wcześniej.

Przykład:

```
SQL> CREATE DATABASE LINK wmstest_link
  2  CONNECT TO app_owner IDENTIFIED BY ********
  3  USING '(DESCRIPTION=
  4            (ADDRESS=(PROTOCOL=TCP)(HOST=ora01)(PORT=1521))
  5            (CONNECT_DATA=(SERVICE_NAME=wmstest))
  6          )';

Database link created.
```

□□ **TCP** = Transmission Control Protocol: protokół sieciowy z potwierdzaniem dostarczenia

□□ **ZMIENIA SYSTEM**

Utworzenie łącza publicznego przez istniejący alias TNS

```
CREATE PUBLIC DATABASE LINK NazwaLinku
CONNECT TO NazwaUsera IDENTIFIED BY HasloUsera
USING 'NazwaInstancji';
```

Jeśli alias `NazwaInstancji` jest już zdefiniowany w `tnsnames.ora` (opisanym na stronie Zmienne środowiskowe, TNS i logowanie do instancji), nie musimy powtarzać całej składni `DESCRIPTION` — samo `USING 'alias'` wystarcza, bo Oracle Net rozwiąże alias tak samo, jak zrobiłby to `sqlplus` przy zwykłym logowaniu.

Przykład:

```
SQL> CREATE PUBLIC DATABASE LINK wmstest
  2  CONNECT TO app_owner IDENTIFIED BY ********
  3  USING 'WMSTEST';

Database link created.
```

Nazwa samego linku najczęściej pokrywa się z nazwą instancji docelowej — nie jest to wymóg silnika, tylko czytelna konwencja, dzięki której `tabela@WMSTEST` mówi wprost, skąd dane pochodzą.

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

□□ **TYLKO ODCZYT**

Sprawdzenie wszystkich zdefiniowanych łączy: ALL_DB_LINKS

```
SELECT db_link, owner, username, host, created FROM all_db_links;
```

> **Pojęcie: ALL_DB_LINKS kontra DBA_DB_LINKS.** `ALL_DB_LINKS` pokazuje linki widoczne dla bieżącego użytkownika — własne prywatne i wszystkie publiczne. `DBA_DB_LINKS` pokazuje wszystkie linki w bazie, niezależnie od właściciela, i wymaga uprawnień administracyjnych do odczytu.

Przykład:

```
DB_LINK              OWNER           USERNAME        HOST         CREATED
-------------------- --------------- --------------- ------------ ---------
WMSTEST                PUBLIC          APP_OWNER       WMSTEST       12-JUN-23
RAPORTY_LINK           APP_USER        RPT_READER      app01    03-MAR-25
```

TYLKO ODCZYT

Sprawdzenie wszystkich łączy w bazie (widok administracyjny): DBA_DB_LINKS

```
SELECT db_link, owner, username, host, created FROM dba_db_links;
```

Przykład:

```
DB_LINK              OWNER           USERNAME        HOST         CREATED
-------------------- --------------- --------------- ------------ ---------
WMSTEST                PUBLIC          APP_OWNER       WMSTEST       12-JUN-23
RAPORTY_LINK           APP_USER        RPT_READER      app01    03-MAR-25
```

TYLKO ODCZYT

Test działania łącza

```
SELECT * FROM dual@NazwaLinku;
```

Najprostszy możliwy test — `dual` istnieje w każdej instancji Oracle, więc to zapytanie sprawdza wyłącznie, czy samo połączenie działa, bez angażowania żadnych danych

aplikacyjnych.

Przykład:

```
SQL> SELECT * FROM dual@wmstest;


D
-
X
```

⚠️ OSTROŻNIE · NIEODWRACALNE

Usunięcie łącza prywatnego

```
DROP DATABASE LINK NazwaLinku;
```

🔒 TYLKO ODCZYT

Usunięcie łącza publicznego

```
DROP PUBLIC DATABASE LINK NazwaLinku;
```

Wariant dla łącza publicznego zamiast prywatnego.

Przykład:

```
SQL> DROP PUBLIC DATABASE LINK wmstest;


Database link dropped.
```

Z życia wzięte

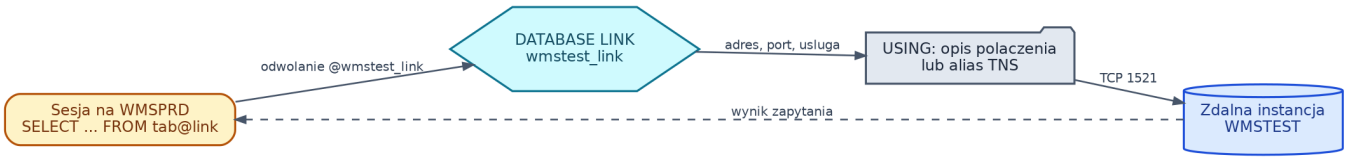
📖 Z ŻYCIA WZIĘTE

Ola potrzebuje raportu łączącego dane WMS z ora01 z tabelą w bazie testowej na innym serwerze. Tomek zakłada `DATABASE LINK wmstest_link` z WMSPRD do WMSTEST i po pierwszym

> teście `SELECT * FROM dual@wmstest_link` raport zaczyna działać. Trzy miesiące później hasło konta docelowego wygasa, a link nadal łączy się starym haszem, więc zapytanie Oli zaczyna zwracać ORA-01017, dopóki ktoś nie usunie linku i nie założy go od nowa.

Czym jest database link

> **Pojęcie: database link.** Obiekt bazy przechowujący dane potrzebne do połączenia się z inną instancją Oracle — adres, port, nazwę usługi i dane logowania — pod jedną, krótką nazwą. Zapytanie `SELECT * FROM tabela@NazwaLinku` łączy się w tle z bazą docelową i wykonuje na niej zapytanie, zwracając wynik tak, jakby tabela była lokalna.



Sesja na WMSPRD odwołuje się do database linku, który przez alias TNS albo opis połączenia dochodzi po sieci do zdalnej instancji WMSTEST (kliknięcie otwiera pełny rozmiar)

> **Pojęcie: link prywatny kontra publiczny.** Link prywatny (`CREATE DATABASE LINK`) widzi i może użyć tylko użytkownik, który go utworzył. Link publiczny (`CREATE PUBLIC DATABASE LINK`) widzi każdy użytkownik bieżącej bazy — wymaga też uprawnienia systemowego `CREATE PUBLIC DATABASE LINK`, szerszego niż zwykłe `CREATE DATABASE LINK`.

Częste błędy

ORA-02019: connection description for remote database not found. Próba użycia linku, który nie istnieje pod tą nazwą dla bieżącego użytkownika — literówka w nazwie linku albo link utworzony jako prywatny przez innego użytkownika, więc niewidoczny dla nas.

ORA-12154: TNS:could not resolve the connect identifier specified. Link utworzony przez `USING 'alias'` odwołuje się do aliasu TNS, którego nie ma w `tnsnames.ora` widocznym z bieżącego `TNS_ADMIN` na serwerze bazy źródłowej — dokładnie ten sam błąd i ta sama przyczyna, co przy zwykłym logowaniu opisanym na stronie Zmienne środowiskowe, TNS i logowanie do instancji, tylko

że tutaj `tnsnames.ora`, które się liczy, jest po stronie serwera wykonującego zapytanie, nie po stronie klienta.

Zmiana hasła użytkownika, przez którego łączy się DB link, bez aktualizacji samego linku. Link trzyma hasło w postaci zaszyfrowanej w momencie utworzenia — zmiana hasła konta docelowego nie propaguje się automatycznie do linku. Naprawa to `DROP` i ponowne `CREATE DATABASE LINK` z nowym hasłem, bo Oracle nie udostępnia polecenia `ALTER DATABASE LINK` zmieniającego dane logowania.

Powiązane: Zmienne środowiskowe, TNS i logowanie do instancji · Użytkownicy, role i uprawnienia

Źródła: notatki własne, dokumentacja Oracle Database SQL Language Reference 19c