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

---

Revision #42
Created 2026-09-24 23:34:21 UTC by Tomasz Konieczny
Updated 2026-10-26 14:08:52 UTC by Tomasz Konieczny