Uprawnienia, role, synonimy – zadania

Zadania należy wykonywać parami, jeden użytkownik działa jako użytkownik A zaś drugi jako użytkownik B. W kilku ćwiczeniach konieczna jest obecność trzeciego użytkownika C.

1. Użytkownik A próbuje odczytać dane z relacji ***pracownicy*** należącej do użytkownika B i *vice versa*.

   ```
   ORA-00942: table or view does not exist
   ```

2. Użytkownik B nadaje użytkownikowi A prawo odczytu danych z własnej relacji ***pracownicy***.

3. Użytkownik A ponownie próbuje odczytać dane z relacji ***pracownicy*** należącej do użytkownika B.

4. Użytkownik A nadaje użytkownikowi B prawo modyfikowania atrybutów *placa_pod* i *placa_dod* we własnej relacji ***pracownicy***.

5. Użytkownik B próbuje zwiększyć o 100% płace podstawowe pracowników w relacji ***pracownicy***, należącej do użytkownika A. Czy może to zrobić? Następnie użytkownik B próbuje ustawić wartość płacy pracownika MORZY w relacji ***pracownicy*** użytkownika A na 2000. Czy ta operacja zakończyła się sukcesem? Ostatecznie użytkownik B próbuje ustawić wszystkim pracownikom w relacji ***pracownicy*** użytkownika A płacę dodatkową na 700. Czy operacja się udała?

6. Użytkownik B tworzy prywatny synonim dla relacji ***pracownicy*** należącej do A i ponawia ostatnią modyfikację z poprzedniego punktu, ustawiając płace dodatkowe na 800, tym razem jednak używając w poleceniu synonimu. Po zakończeniu modyfikacji użytkownik B zatwierdza swoje zmiany poleceniem COMMIT.

7. Użytkownik B próbuje odczytać dokonane przez siebie zmiany w relacji ***pracownicy*** użytkownika A.

8. Użytkownicy A i B oglądają informacje ze słownika bazy danych dotyczące przyznanych uprawnień obiektowych:

   ```
   select owner, table_name, grantee, grantor, privilege
   from   user_tab_privs;

   select table_name, grantee, grantor, privilege
   from   user_tab_privs_made;
   ```

```
select owner, table_name, grantor, privilege
from   user_tab_privs_recd;

select owner, table_name, column_name, grantee, grantor, privilege
from   user_col_privs;

select table_name, column_name, grantee, grantor, privilege
from   user_col_privs_made;

select owner, table_name, column_name, grantor, privilege
from   user_col_privs_recd;
```

9. Użytkownik A odbiera użytkownikowi B prawo modyfikacji własnej relacji pracownicy. B następnie próbuje modyfikować (bezpośrednio i za pomocą synonimu) relację ***pracownicy*** należącą do A.

10. Użytkownicy tworzą role i nadają tym rolom prawo odczytu i modyfikowania danych we własnych relacjach ***pracownicy***. Rola użytkownika A powinna być chroniona hasłem, rola użytkownika B jest rolą bez hasła. Nazwy ról powinny zostać skonstruowane przez dodanie do słowa *ROLA_* numeru indeksu studenta, np. *ROLA_12345* dla użytkownika *INF12345*.

11. Użytkownik A nadaje stworzoną przez siebie rolę użytkownikowi B. B próbuje odczytać zawartość relacji ***pracownicy*** należącej do A.

12. Użytkownik B włącza rolę przyznaną mu przez użytkownika A. B próbuje odczytać zawartość relacji ***pracownicy*** należącej do A. B przegląda informacje ze słownika bazy danych dotyczącą uprawnień związanych z rolami.

```
select granted_role, admin_option from user_role_privs
where  username = 'B';

select role, owner, table_name, column_name, privilege
from   role_tab_privs;
```

13. Użytkownik A odbiera użytkownikowi B rolę. Użytkownik B próbuje odczytać informacje z relacji ***pracownicy*** należącej do użytkownika A.

14. Użytkownik B odłącza się od bazy danych, przyłącza ponownie i próbuje dokonać odczytu danych z relacji ***pracownicy*** użytkownika A.

15. Użytkownik A próbuje dokonać modyfikacji danych relacji ***pracownicy*** użytkownika B.

16. Użytkownik B nadaje użytkownikowi A rolę, jaką utworzył w p. 10. Użytkownik A ponownie próbuje dokonać modyfikacji danych relacji ***pracownicy*** użytkownika B.

17. Użytkownik A odłącza się od bazy danych, przyłącza ponownie i znowu próbuje dokonać modyfikacji danych relacji ***pracownicy*** użytkownika B.

18. Użytkownik B odbiera swojej roli prawo modyfikowania własnej relacji ***pracownicy***. Użytkownik A ponownie próbuje dokonać modyfikacji danych relacji ***pracownicy*** należącej do użytkownika B.

19. Obaj użytkownicy usuwają utworzone przez siebie role.

20. Użytkownik A nadaje użytkownikowi B prawo odczytu własnej relacji ***pracownicy*** wraz z prawem dalszego przyznawania tego uprawnienia. Użytkownik B przekazuje to prawo dalej użytkownikowi C. Użytkownik C próbuje odczytać dane z relacji ***pracownicy*** należącej do użytkownika A.

21. Wszyscy użytkownicy (A, B i C) oglądają zawartość słownika bazy danych dotyczącą nadanych i otrzymanych uprawnień obiektowych.

22. Użytkownik A próbuje odebrać uprawnienia do swojej relacji ***pracownicy*** użytkownikowi C. Następnie próbuje odebrać te same uprawnienia użytkownikowi B. Użytkownicy raz jeszcze oglądają słownik bazy danych.

23. Użytkownik A tworzy perspektywę ***prac20*** udostępniającą nazwiska i płace pracowników zespołu 20. Następnie przenosi całość uprawnień do odczytu i modyfikacji z relacji ***pracownicy*** na utworzoną perspektywę ***prac20***. Użytkownik B modyfikuje relację ***pracownicy*** użytkownika A za pomocą udostępnionej mu perspektywy.

24. Użytkownik A tworzy w swoim schemacie funkcję PL/SQL o nazwie ***funLiczEtaty***, która policzy liczbę rekordów relacji ***etaty*** i zwróci ją jako wynik. Następnie nadaje prawo wykonywania tej funkcji użytkownikowi B.

25. Użytkownik B wykonuje funkcję ***funLiczEtaty***, a następnie próbuje zweryfikować poprawność jej wyniku licząc za pomocą zapytania SQL rekordy w relacji ***etaty*** użytkownika A.

26. Użytkownik A ponownie tworzy funkcję ***funLiczEtaty***, jednak teraz funkcja ma działać z prawami użytkownika bieżącego (wykonującego) – z klauzulą `AUTHID CURRENT_USER`. Następnie ponownie nadaje prawo wykonywania tej funkcji użytkownikowi B.

27. Użytkownik B ponownie wykonuje funkcję ***funLiczEtaty***. Czy otrzymany wynik różni się od wyniku wykonania funkcji z punktu 25.?

28. Użytkownik A dodaje do swojej relacji ***etaty*** nowy rekord, opisujący etat o nazwie WYKŁADOWCA i pensji od 1000 do 2000 zł. Po dodaniu rekordu A zatwierdza operację poleceniem COMMIT.

29. Użytkownik B ponownie wykonuje funkcję ***funLiczEtaty***. Dlaczego otrzymany wynik nie różni się wyniku wykonania funkcji z punktu 27.?

30. Użytkownik B tworzy relację ***test*** o schemacie: *id* number(2), *tekst* varchar2(20) i dodaje do niej dwa rekordy: (1,'pierwszy'), (2, 'drugi'). Następnie tworzy procedurę ***procPokazTest***, której zadaniem jest wypisanie na konsoli zawartości kolumny *tekst* ze wszystkich rekordów relacji ***test***. (uwaga: odwołując się do relacji ***test*** w ciele procedury użytkownik B **NIE** poprzedza jej nazwy swoją nazwą użytkownika). Procedura ma działać z uprawnieniami bieżącego użytkownika (klauzula `AUTHID CURRENT_USER`). Następnie B nadaje użytkownikowi A:
    a. prawo wykonywania procedury ***procPokazTest*** oraz
    b. prawo odczytu relacji ***test***,

31. Użytkownik A próbuje uruchomić procedurę ***procPokazTest*** użytkownika B. Dlaczego operacja kończy się niepowodzeniem? Co może zrobić użytkownik A, aby mógł wykonać procedurę ***procPokazTest***?

32. Utwórz relację ***info_dla_znajomych*** o poniższym schemacie:

```
NAZWA VARCHAR2(20) NOT NULL
INFO VARCHAR2(200) NOT NULL
```

    Wpisz do relacji kilka krotek. Jako wartości atrybutu *nazwa* podaj identyfikatory innych użytkowników w bazie danych. Utwórz perspektywę ***info4u*** i nadaj do niej odpowiednie prawa w ten sposób, aby każdy użytkownik bazy danych mógł odczytać z perspektywy ***info4u*** informacje przeznaczone tylko i wyłącznie dla siebie.