12. LABORATORIJSKE VJEŽBE

Tema: Rukovanje usllvima u pohranjenim programima

Zadatak 1.

a) **Šta** će se dogoditi i **zašto** pri izvođenju procedure **test1**?

```
CREATE PROCEDURE test1 ()
BEGIN
  DECLARE kod CHAR(5);
  DECLARE poruka TEXT;
  DECLARE rezultat TEXT;
  DECLARE CONTINUE HANDLER FOR SQLEXCEPTION
    BEGIN
      GET DIAGNOSTICS CONDITION 1
      kod = RETURNED_SQLSTATE, poruka = MESSAGE_TEXT;
      SET rezultat = CONCAT('greska = ',kod,', poruka = ',poruka);
    END;
   BEGIN
     SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'Pogreska 1000',
MYSQL_ERRNO = 1000;
     SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'Pogreska 1001',
MYSQL_ERRNO = 1001;
   END;
   SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'Pogreska 1002',
MYSQL_ERRNO = 1002;
  SELECT rezultat;
END//
```

Šta će se dogoditi i **zašto** pri izvođenju procedure **test1**, ako se HANDLER deklariše sa akcijom EXIT?

b) **Šta** će se dogoditi i **zašto** pri izvođenju procedure **test2**?

```
CREATE PROCEDURE test2 ()
BEGIN
  DECLARE kod CHAR(5);
  DECLARE poruka TEXT;
  DECLARE rezultat TEXT;
   BEGIN
    DECLARE CONTINUE HANDLER FOR SQLEXCEPTION
    BEGIN
      GET DIAGNOSTICS CONDITION 1
      kod = RETURNED_SQLSTATE, poruka = MESSAGE_TEXT;
      SET rezultat = CONCAT('greska = ',kod,', poruka = ',poruka);
    END;
     SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'Pogreska 1000',
MYSQL_ERRNO = 1000;
    SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'Pogreska 1001',
MYSQL_ERRNO = 1001;
   END;
  SELECT rezultat;
END//
```

Šta će se dogoditi i **zašto** pri izvođenju procedure **test1**, ako se HANDLER deklariše sa akcijom EXIT?

c) **Šta** će se dogoditi i **zašto** pri izvođenju procedure **test3**?

```
CREATE PROCEDURE test3 ()
BEGIN
  DECLARE kod CHAR(5);
  DECLARE poruka TEXT;
  DECLARE rezultat TEXT;
  DECLARE CONTINUE HANDLER FOR SQLSTATE '10003'
  BEGIN
    GET DIAGNOSTICS CONDITION 1
    kod = RETURNED_SQLSTATE;
    SET rezultat = CONCAT('greska = ',kod,', poruka = Druga');
  END;
  DECLARE CONTINUE HANDLER FOR SQLSTATE '10002'
  BEGIN
    GET DIAGNOSTICS CONDITION 1
    kod = RETURNED_SQLSTATE;
    SET rezultat = CONCAT('greska = ',kod,', poruka = Prva');
  END;
  DECLARE CONTINUE HANDLER FOR SQLSTATE '10001'
  BEGIN
    GET DIAGNOSTICS CONDITION 1
    kod = RETURNED_SQLSTATE;
    SET rezultat = CONCAT('greska = ',kod,', poruka = Treca');
  END;
  BEGIN
    DECLARE CONTINUE HANDLER FOR SQLSTATE '10001'
    BEGIN
      SIGNAL SQLSTATE '10002';
    END;
    DECLARE CONTINUE HANDLER FOR SQLEXCEPTION
    BEGIN
      SIGNAL SQLSTATE '10003';
    END;
    SIGNAL SQLSTATE '10001';
  END;
  SELECT rezultat;
END//
```

d) **Šta** će se dogoditi i **zašto** pri izvođenju procedure **test3**, ako se izbaci deklaracija rukovatelja za `SQLSTATE '10001` iz ugniježdenog BEGIN ... END bloka?

e) **Šta** će se dogoditi i **zašto** pri izvođenju procedure **test3**, ako se naredba `SIGNAL SQLSTATE '10001';` premjesti neposredno ispred naredbe SELECT?

Tema 2: (Zaštita od neovlaštenog pristupa)

Zadatak 1.

Napisati niz SQL naredbi za dodjeljivanje dozvola u bazi podataka, pomoću kojih će se korisniku "haso" (koji nije vlasnik dotične baze podataka) omogućiti:

A. pregled svih podataka u relacijama *mjesto, zupanija, pred, nastavnik* i *orgjed*
B. pregled svih podataka u relaciji *stud*, osim datuma rođenja
C. pregled svih podataka u relaciji *ispit*, ali samo za ispite na kojima je dobivena pozitivna ocjena
D. pregled ukupnog broja negativnih ocjena za svaki predmet. Korisnik pri tome ne smije dobiti mogućnost pregledavanja pojedinačnih zapisa u relaciji ispit (onih zapisa za koje je upisana negativna ocjena)
E. omogućiti izračunavanje ukupne ocjene obrane za zadani matični broj studenta i datum prijave diplomskog ispita. Ukupna ocjena obrane računa se uz pomoć procedure *ocjObrane*
F. omogućiti unos, izmjenu i brisanje svih podataka u relaciji *mjesto*
G. omogućiti izmjenu podatka *nazZupanija* u relaciji *zupanija*
H. omogućiti unos, izmjenu i brisanje podataka u relaciji *nastavnik*, ali samo za nastavnike čija organizacijska jedinica ima šifru 100006

Ukoliko je potrebno, treba kreirati i dodatne objekte (npr. pogled-view). Korisnik "haso" treba dobiti sve navedene dozvole, ali ništa više od toga (npr. korisnik "haso" nema dozvolu za pregled podataka u *dipkom* i *diplom* relacijama).
Na laboratorijskim vježbama podijelite se u parove (osoba A, osoba B). Osoba A će u svojoj bazi podataka izvesti pripremljene naredbe za dodjeljivanje dozvola osobi B. U pripremljenim naredbama treba promijeniti naziv korisnika (haso) u stvarni naziv korisnika kojem dodjeljujete dozvole (osobi B).
Osoba B će nakon toga pristupiti bazi podataka za koju je dobila dozvole i testirati da li su dozvole koje je dobila u skladu sa specifikacijama iz zadatka.
Osoba A i osoba B nakon toga zamjenjuju uloge i ponavlja se cijeli postupak.

Zadatak 2.
Procedura *azurDiplom* koju ste napisali na prethodnim laboratorijskim vježbama sadrži jedan bitan nedostatak - ukoliko se za vrijeme obavljanja procedure dogodi pogreška, transakcija, koju je procedura sama pokrenula, ostaje aktivna. Potrebno je (koristeći kao osnovu proceduru *azurDiplom*) napisati proceduru *boljaAzurDiplom* koja će imati sljedeća svojstva (svojstva A, B i C):

A. Ukoliko se prilikom postavljanja READ ili PROMOTABLE ključa u relaciji *diplom* ili *dipkom* dogodi da se ključ ne može postaviti, procedura mora poništiti transakciju i dojaviti pogrešku -746, 0, "Relacija diplom/dipkom: READ ili PROMOTABLE LOCK nije odobren"

B. Ukoliko se prilikom postavljanja WRITE ključa u relaciji *diplom* dogodi da se ključ ne može postaviti, procedura mora poništiti transakciju i dojaviti pogrešku -746, 0, "Relacija diplom: WRITE LOCK nije odobren"

Smatramo da je jedna od ove dvije pogreške detektirana ukoliko se pri obavljanju procedure *ocjObrane* ili pri obavljanju UPDATE operacije **nad relacijom *diplom*** pojavi neka od pogrešaka:

```
-1027 error: '%s' is locked against change
-1099 error: Table '%s' was locked with a READ lock and can't be updated
-1205 error: Lock wait timeout exceeded; try restarting transaction
-1213 error: Deadlock found when trying to get lock; try restarting
transaction
-1223 error: Can't execute the query because you have a conflicting read lock
-1614 error: Transaction branch was rolled back: deadlock was detected
-1689 error: Wait on a lock was aborted due to a pending exclusive lock
```

Pogreška će se dogoditi npr. ako prije obavljanja procedure *boljaAzurDiplom*, neki drugi proces izvede naredbe:

```
start transaction;
update dipkom set sifNastavnik = sifNastavnik
```

Pogreška će se dogoditi npr. ako prije obavljanja procedure *boljaAzurDiplom*, neki drugi proces izvede naredbe

```
start transaction;
set transaction isolation level serializable;
select * from diplom
   where mbrStud = 1133
```

C. Ukoliko se unutar procedure dogodi bilo koja druga pogreška, transakcija se mora poništiti i pozivajućem programu proslijediti opis originalne pogreške koja je prouzročila iznimku.

Neka "nepredviđena" pogreška će se dogoditi npr. ako prije obavljanja procedure *boljaAzurDiplom*, neki drugi proces izvede naredbe:

```
rename table dipkom to dipkom2
```

Obavljanjem navedenih naredbi preimenovat će se relacija dipkom ("relacija dipkom tada više neće postojati"). Budući da u proceduri nije predviđen mehanizam za rukovanje takvom pogreškom, dogodit će se "neočekivana" iznimka:

```
Error code: 1146 Table stusluEmMe.dipkom does not exists.
```

Procedura mora ovu pogrešku proslijediti u pozivajući program (u ovom slučaju MySQL Workbranch)

Na laboratorijskim vježbama testirati ponašanje pripremljene procedure u slučajevima pojave pogrešaka opisanih pod A, B i C.