SQL DDL - Pohranjene procedure

Evidencija račun

Potrebno je evidentirati račune izdane u trgovini. Za svaki račun potrebno je pratiti *broj_računa*, *datum_izdavanja* i *zaposlenika* koji je račun izdao, te kupca koji je platio račun. Za zaposlenika se prati: *ime*, *prezime*, *oib*, *datum_zaposlenja*, dok se za kupca prati *ime* i *prezime*. Na svakom računu se nalazi stavke koje u sebi sadrže *artikl* i *količinu*. Artikl se sastoji od *naziva* i *cijene*.

```
kupac(id, ime, prezime)
zaposlenik(id, ime, prezime, oib, datum_zaposlenja)
artikl(id, naziv, cijena)
racun(id, id_zaposlenik, id_kupac, broj, datum_izdavanja)
stavka_racun(id, id_racun, id_artikl, kolicina)
```

Baza podataka & tablice

```
CREATE DATABASE trgovina;
USE trgovina;

CREATE TABLE kupac (
  id INTEGER NOT NULL,
  ime VARCHAR(10) NOT NULL,
  prezime VARCHAR(15) NOT NULL,
  PRIMARY KEY (id)
);

CREATE TABLE zaposlenik (
  id INTEGER NOT NULL,
  ime VARCHAR(10) NOT NULL,
  prezime VARCHAR(15) NOT NULL,
  oib CHAR(11) NOT NULL,
  datum_zaposlenja DATETIME NOT NULL,
  PRIMARY KEY (id)
);

CREATE TABLE artikl (
  id INTEGER NOT NULL,
  naziv VARCHAR(20) NOT NULL,
  cijena NUMERIC(10,2) NOT NULL,
  PRIMARY KEY (id)
);
```

```sql
CREATE TABLE racun (
  id INTEGER NOT NULL,
  id_zaposlenik INTEGER NOT NULL,
  id_kupac INTEGER NOT NULL,
  broj VARCHAR(100) NOT NULL,
  datum_izdavanja DATETIME NOT NULL,
  PRIMARY KEY (id),
  FOREIGN KEY (id_zaposlenik) REFERENCES zaposlenik (id),
  FOREIGN KEY (id_kupac) REFERENCES kupac (id)
);

CREATE TABLE stavka_racun (
  id INTEGER NOT NULL,
  id_racun INTEGER NOT NULL,
  id_artikl INTEGER NOT NULL,
  kolicina INTEGER NOT NULL,
  PRIMARY KEY (id),
  FOREIGN KEY (id_racun) REFERENCES racun (id) ON DELETE CASCADE,
  FOREIGN KEY (id_artikl) REFERENCES artikl (id),
  UNIQUE (id_racun, id_artikl)
);
```

Unos podataka

```sql
INSERT INTO kupac VALUES (1, 'Lea', 'Fabris'),
                         (2, 'David', 'Sirotić'),
                         (3, 'Tea', 'Bibić');

INSERT INTO zaposlenik VALUES
  (11, 'Marko', 'Marić', '123451', STR_TO_DATE('01.10.2020.', '%d.%m.%Y.')),
  (12, 'Toni', 'Milovan', '123452', STR_TO_DATE('02.10.2020.', '%d.%m.%Y.')),
  (13, 'Tea', 'Marić', '123453', STR_TO_DATE('02.10.2020.', '%d.%m.%Y.'));

INSERT INTO artikl VALUES (21, 'Puding', 5.99),
                          (22, 'Milka čokolada', 30.00),
                          (23, 'Čips', 9);

INSERT INTO racun VALUES
  (31, 11, 1, '00001', STR_TO_DATE('05.10.2020.', '%d.%m.%Y.')),
  (32, 12, 2, '00002', STR_TO_DATE('06.10.2020.', '%d.%m.%Y.')),
  (33, 12, 1, '00003', STR_TO_DATE('06.10.2020.', '%d.%m.%Y.'));

INSERT INTO stavka_racun VALUES (41, 31, 21, 2),
                                (42, 31, 22, 5),
                                (43, 32, 22, 1),
                                (44, 32, 23, 1);
```

Definiranje procedure: Procedura će pri pozivu povećati cijenu artikala za vrijednost parametra *postotak*:

```
DELIMITER //
CREATE PROCEDURE povecaj_cijene_artikala(postotak DECIMAL(8, 2))
BEGIN

  UPDATE artikl
    SET cijena = cijena * (1 + postotak/100);

END //
DELIMITER ;


-- Ispis
SELECT * FROM artikl;

-- Pozivanje procedure
CALL povecaj_cijene_artikala(10);

-- Ispis
SELECT * FROM artikl;
```

IN/OUT parametri: Procedura u varijablu *rezultat* sprema zbroj vrijednosti parametar *a* i *b*:

```
DELIMITER //
CREATE PROCEDURE zbroji(IN a INTEGER, IN b INTEGER, OUT rezultat INTEGER)
BEGIN

  SET rezultat = a + b;

END //
DELIMITER ;


CALL zbroji(1, 3, @rez);
SELECT @rez FROM DUAL;
```

INOUT parametar: Procedura povećava vrijednost parametra *br* za 1:

```
DELIMITER //
CREATE PROCEDURE povecaj_brojac(INOUT br INTEGER)
BEGIN

    SET br = br + 1;

END //
DELIMITER ;


SET @brojac = 5;
CALL povecaj_brojac(@brojac);
SELECT @brojac FROM DUAL;
```

Implicitni kursor: Procedura dohvaća minimalnu i maksimalnu cijenu u varijable *min_cijena* i *max_cijena*:

```
DELIMITER //
CREATE PROCEDURE dohvati_info_o_cijeni_artikala(OUT min_cijena DECIMAL(10, 2),
OUT max_cijena DECIMAL(10, 2))
BEGIN

  SELECT MIN(cijena), MAX(cijena) INTO min_cijena, max_cijena
    FROM artikl;

END //
DELIMITER ;


CALL dohvati_info_o_cijeni_artikala(@min_cijena, @max_cijena);
SELECT @min_cijena, @max_cijena FROM DUAL;
```

Eksplicitni kursor: Procedura dohvaća minimalnu i maksimalnu cijenu u varijable *min_cijena* i *max_cijena* korištenjem eksplicitnog kursora:

```
DELIMITER //
CREATE PROCEDURE dohvati_info_o_cijeni_artikala_2(OUT min_cijena DECIMAL(10,
2), OUT max_cijena DECIMAL(10, 2))
BEGIN
  DECLARE cur CURSOR FOR
  SELECT MIN(cijena), MAX(cijena) FROM artikl;

  OPEN CUR;
  FETCH cur INTO min_cijena, max_cijena;
  CLOSE cur;

END //
DELIMITER ;


CALL dohvati_info_o_cijeni_artikala_2(@min_cijena, @max_cijena);
SELECT @min_cijena, @max_cijena FROM DUAL;
```