DDL - Ponavljanje

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

```
DROP DATABASE trgovina;
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)
);
```

```sql
CREATE TABLE artikl (
  id INTEGER NOT NULL,
  naziv VARCHAR(20) NOT NULL,
  cijena NUMERIC(10,2) NOT NULL CHECK (cijena > 0),
  PRIMARY KEY (id)
);

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

```
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);
```

Imenovanje ograničenja: Svako ograničenje možemo imenovati kako bismo lakše pronašli razlog greške u slučaju kada se ograničenje prekrši

```
CREATE TABLE kupac (
  ...
  CONSTRAINT kupac_pk PRIMARY KEY (id)
);

CREATE TABLE zaposlenik (
  ...
  CONSTRAINT kupac_pk PRIMARY KEY (id)
);

CREATE TABLE artikl (
  ..
  CONSTRAINT artikl_pk PRIMARY KEY (id),
  CONSTRAINT artikl_cijena_ck CHECK (cijena > 0)
);

CREATE TABLE racun (
  ...
  CONSTRAINT racun_pk PRIMARY KEY (id),
  CONSTRAINT racun_zaposlenik_fk FOREIGN KEY (id_zaposlenik) REFERENCES
zaposlenik (id),
  CONSTRAINT racun_kupac_fk FOREIGN KEY (id_kupac) REFERENCES kupac (id)
);

CREATE TABLE stavka_racun (
  ...
  CONSTRAINT stavka_fk PRIMARY KEY (id),
  CONSTRAINT stavka_racun_fk FOREIGN KEY (id_racun) REFERENCES racun (id) ON
DELETE CASCADE,
  CONSTRAINT stavka_artikl_fk FOREIGN KEY (id_artikl) REFERENCES artikl (id),
  CONSTRAINT stavka_uk UNIQUE (id_racun, id_artikl)
);

# Testiranje poruke greške
INSERT INTO artikl VALUES (24, 'Fanta', -1);
```

Datumski i vremenski tipovi podatka: Pregled datumskih i vremenskih tipova podataka i funkcija:

```
# Kreiranje testne tablice
CREATE TABLE test_datum_vrijeme (
    datum DATE,
    vrijeme TIME,
    vrijeme_p TIME(5) # Broj 5 znači da će se nakon sekundi spremati još 5
brojeva
);

# Funkcija NOW() dohvaća trenutni datum i vrijeme
SELECT NOW() FROM DUAL;

# Primijetimo da se javlja upozorenje kod unosa polja 'datum'
INSERT INTO test_datum_vrijeme VALUES (NOW(), NOW(), NOW());
INSERT INTO test_datum_vrijeme VALUES (NOW(), NOW(), NOW());

# Sve je ipak dobro spremljeno
SELECT * FROM test_datum_vrijeme;

# Funkcija DATE izvlači samo podatak o datumu, dok funkcija TIME samo podatak o
vremenu
SELECT DATE(NOW()), TIME(NOW()) FROM DUAL;
# Funkcije CURRENT_DATE i CURRENT_TIME su samo skraćeni način za postizanje
prethodnog rezultata
SELECT CURRENT_DATE(), CURRENT_TIME() FROM DUAL;

# U oba slučaja unos prolazi bez greške i upozorenja
INSERT INTO test_datum_vrijeme VALUES (DATE(NOW()), TIME(NOW()), TIME(NOW()));
INSERT INTO test_datum_vrijeme VALUES (CURRENT_DATE(), CURRENT_TIME(),
CURRENT_TIME());

SELECT * FROM test_datum_vrijeme;
```

Tipovi podataka za datum i vrijeme: Tipovi podatka za istodobno spremanje datuma i vremena

```
CREATE TABLE test_datum_i_vrijeme (
    datum_datetime DATETIME,
    datum_datetime_p DATETIME(5),
    datum_timestamp TIMESTAMP,
    datum_timestamp_p TIMESTAMP(5)
);

INSERT INTO test_datum_i_vrijeme
  VALUES (NOW(), NOW(), NOW(), NOW());

# u MySQL je funkcija CURRENT_TIMESTAMP sinonim za NOW()
INSERT INTO test_datum_i_vrijeme
  VALUES (CURRENT_TIMESTAMP(), CURRENT_TIMESTAMP(), CURRENT_TIMESTAMP(),
CURRENT_TIMESTAMP());

SELECT * FROM test_datum_i_vrijeme;
```

Rad sa datumskim vrijednostima:

```
# Priprema
CREATE TABLE datum_vrijeme (
    datum_time TIME,
    datum_date DATE,
    datum_datetime DATETIME
);

INSERT INTO datum_vrijeme VALUES (TIME(NOW()), DATE(NOW()), NOW());
INSERT INTO datum_vrijeme VALUES (TIME(NOW()), DATE(NOW()), NOW());

SELECT * FROM datum_vrijeme;


# Izvlačenje određenih vrijednosti
SELECT EXTRACT(MINUTE FROM datum_time),
       EXTRACT(HOUR FROM datum_date),
       EXTRACT(MONTH FROM datum_datetime)
  FROM datum_vrijeme;

# U MySQL-u postoje i funkcije kao što su MINUTE, HOUR i sl. za istu. namjenu
SELECT MINUTE(datum_time),
       HOUR(datum_date),
       MONTH(datum_datetime)
  FROM datum_vrijeme;


# Unos datuma dobivenih pretvaranjem izraza pomoću zadanog formata
INSERT INTO datum_vrijeme (datum_time, datum_date, datum_datetime)
  VALUES (
      TIME(STR_TO_DATE('01.10.2020. 22:10:55', '%d.%m.%Y. %H:%i:%s')),
      STR_TO_DATE('01.10.2020.', '%d.%m.%Y.'),
      STR_TO_DATE('01.10.2020. 22:10:55', '%d.%m.%Y. %H:%i:%s')
  );


SELECT * FROM datum_vrijeme;


# Formatiranje datuma prilikom ispisa
SELECT DATE_FORMAT(datum_time, '%H:%i:%s'),
       DATE_FORMAT(datum_date, '%d.%m.%Y. %H:%i:%s'),
       DATE_FORMAT(datum_datetime, '%a, %b, %c - %H:%i')
  FROM datum_vrijeme;
```

Interval: Možemo koristiti INTERVAL kako bismo promijenili datum ili vrijeme za željeni interval

```
# Primjer
SELECT NOW() + INTERVAL 1 YEAR FROM DUAL;

# Primjer
SELECT datum_time + INTERVAL 1 MINUTE,
       datum_date + INTERVAL 10 DAY,
       datum_datetime - INTERVAL 60 SECOND
  FROM datum_vrijeme
UNION
SELECT NULL, NULL, NULL FROM DUAL
UNION
SELECT datum_time, datum_date, datum_datetime
  FROM datum_vrijeme;
```

Zadatak: Prikaži sve račune koji su izdani u posljednjih godinu dana (unatrag godinu dana od današnjeg dana)

```
SELECT *
  FROM racun
  WHERE datum_izdavanja > NOW() - INTERVAL 1 YEAR;
```

CAST: Pretvaranje tipova podataka

```
SELECT 4 + '1.01' FROM DUAL;

SELECT 4 + CAST('1.01' AS DECIMAL(2, 1)) FROM DUAL;

SELECT 4 + CAST('treba ipak paziti na podatak koji se pretvara' AS DECIMAL(3,
2))
  FROM DUAL;
```

Izmjena sheme tablice:

```
# Preimenovanje tablice
ALTER TABLE artikl
  RENAME TO proizvod;

# Dodavanje novog stupca
ALTER TABLE proizvod
  ADD COLUMN jedinica_mjere VARCHAR(10) DEFAULT 'lit';

# Mijenjanje postojećeg stupca
ALTER TABLE proizvod
  MODIFY COLUMN jedinica_mjere VARCHAR(5) NOT NULL DEFAULT 'kom';

# Dodavanje ograničenja
ALTER TABLE proizvod
  ADD CONSTRAINT proizvod_jm_ck CHECK (jedinica_mjere IN ('kom', 'kg', 'lit'));

# Brisanje ograničenja
ALTER TABLE proizvod
  DROP CONSTRAINT proizvod_jm_ck;

# Brisanje stupca
ALTER TABLE proizvod
  DROP COLUMN jedinica_mjere;

# Ispis
SELECT * FROM proizvod;

# Vraćamo natrag naziv tablice
ALTER TABLE proizvod
  RENAME TO artikl;
```

Privremene tablice:

```
# Kada sesija završi, privremena tablica se briše
CREATE TEMPORARY TABLE prihod_na_datum(
  datum DATETIME,
  iznos DECIMAL(20, 2)
);

# Unos podataka kao kod normalne tablice
INSERT INTO prihod_na_datum VALUES (NOW(), 1000.00);

# Ispis podataka kao kod normalne tablice
SELECT * FROM prihod_na_datum;

# Možemo napraviti unos podataka koji su rezultat upita (vrijedi i za normalne
tablice)
INSERT INTO prihod_na_datum
  SELECT datum_izdavanja,
       SUM(cijena * kolicina) AS iznos
    FROM racun r
    INNER JOIN stavka_racun s ON s.id_racun = r.id
    INNER JOIN artikl a ON s.id_artikl = a.id
    GROUP BY datum_izdavanja;


# Možemo napraviti novu privremenu tablicu iz rezultata upita
CREATE TEMPORARY TABLE tmp_artikli AS (SELECT * FROM artikl);
SELECT * FROM tmp_artikli;

# Vrijedi i za normalnu tablicu
CREATE TABLE jeftini_artikli AS (SELECT * FROM artikl WHERE cijena < (SELECT
AVG(cijena) FROM artikl));
SELECT * FROM jeftini_artikli;

# Trik: ako ne želimo da se automatski kopiraju i podaci
CREATE TABLE prazna_tablica AS (SELECT * FROM artikl LIMIT 0);
SELECT * FROM prazna_tablica;
```

Zadaća

1. Prikaži sve zaposlenike sa dodatnim stupcem (sa nazivom *'uljepsani_datum'*) koji će prikazivati datum zaposlenja u obliku sljedećeg primjera: **zaposlen u 10:30 sati, datuma: 01.10.2020.**
2. Prikaži sve zaposlenike koji su zaposleni u posljednjih godinu i pol dana
3. Napiši naredbu koje će tablici *'zaposlenik'* dodati stupac sa nazivom *'placa'* sa zadanom vrijednosti od 5000
4. Napiši naredbu koja će u tablicu *'zaposlenik'* dodati ograničenje sa pripadnim nazivom, koji će provjeravati da plaća zaposlenika mora biti iznad 3000
5. Napravi privremenu tablicu sa nazivom *'najskuplji_artikl'* koja će imati iste stupce kao i tablica *'artikl'*, dok je u nju potrebno spremiti artikl sa najvećom cijenom (rezultat: privremena tablica treba sadržavati redak: *22, 'Milka čokolada', 30.00*)