Elektrotehnički fakultet Sarajevo
Odsjek za računarstvo i informatiku

Zadaća 2

VAŽNO: FORMAT ZADAĆE PAŽLJIVO PROČITATI

Zadaci zahtijevaju da popunite c2 kviz i pošaljete na zamger text SQL upita. Sve SQL upite i ostale SQL skripte treba sastaviti u jedan tekst i ubaciti u tekstualno polje zadatka na zamgeru. Primjer izgleda upita:

```
--1.
Select * from drzava;
--2.
Select * from drzava;
--3.
--4.
--5.
--6.
--7.
--8.
--9.
--10.
```

Da bi zadatak bio bodovan, potrebno je da je popunjen c2 dio i poslan zamger dio. Odgovaranje na pitanje c2 dijela bez zamger dijela će biti kažnjavano negativnim poenima. Eventualna neslaganja između odgovora na c2 i sadržaja poslanih datoteka na zamgeru neće biti naknadno bodovana, pa dobro provjerite da ste poslali željene odgovore nakon konačnog slanja na c2! Poslije slanja dobro provjeriti da li su sve datoteke ispravno poslane. Slanje zadaće van roka se neće uzeti u razmatranje. Zadaću je moguće popuniti-slati više puta i rok je dovoljno dug tako da nema potrebe čekati 23:58 da se pošalje zadaća. Pošaljite je dan ranije kako bi stigli ispraviti sve eventualne tehničke poteškoće na vrijeme. Rok zadaće je objavljen na informacionom sistemu zamger.

Zadatak 1 (5b)

Ako se prebacite na schema Erd komandom ALTER SESSION SET CURRENT_SCHEMA = erd; pronaći ćete bazu podataka za prodaju proizvoda. Za datu bazu podataka poznato je da ima tabele: *KONTINENT, DRZAVA, GRAD, LOKACIJA, FIZICKO_LICE, PRAVNO_LICE, PROIZVODJAC, KURIRSKA_SLUZBA, UGOVOR_ZA_PRAVNO_LICE, KUPAC, UPOSLENIK, ODJEL, UGOVOR_ZA_UPOSLENIKA, SKLADISTE, KATEGORIJA, POPUST, PROIZVOD, KOLICINA, GARANCIJA, ISPORUKA, NARUDZBA_PROIZVODA i FAKTURA*
i poznato je da:
PROIZVODJAC i KURIRSKA_SLUZBA imaju primary key koji je ujedno i foreign key na PRAVNO_LICE
KUPAC i UPOSLENIK imaju primary key koji je ujedno i foreign key na FIZICKO_LICE.

Napisati upit koji će:

1. Ispisati nazive bez ponavljanja svih pravnih lica za koje postoji fizičko lice na istoj lokaciji.
2. Ispisati bez ponavljanja datum potpisivanja ugovora (u formatu dd.MM.yyyy) i naziv

pravnog lica za sve ugovore kod kojih je datum potpisivanja poslije prvog datuma kupoprodaje faktura koje sadrže proizvod kod kojeg broj mjeseci garancije nije null.

3. Ispisati nazive proizvoda čija je kategorija jednaka bar jednoj kategoriji proizvoda čija je ukupna količina jednaka maksimalnoj od ukupnih količina svakog proizvoda posebno.
4. Ispisati nazive proizvoda i nazive proizvođača za sve proizvode za čijeg proizvođača postoji proizvod čija je cijena proizvoda veća od prosjeka cijena svih proizvoda.
5. Ispisati ime i prezime kupaca koji su istovremeno uposlenici i sumu potrošenog iznosa na fakture koje su platili za svakog od njih (grupisani po imenu PA prezimenu) čija je suma iznosa veća od prosjeka (zaokruženog na dvije decimale) suma iznosa faktura fizičkih lica (grupisanih po imenu PA prezimenu).
6. Ispisati naziv kurirske službe čija je suma količine jednog proizvoda u njenim narudžbama gdje ima popusta jednaka maksimalnoj sumi količine jednog proizvoda u narudžbama svih kurirskih službi (suma grupisano po id kurirske službe) gdje ima popusta.
7. Ispisati ime i prezime svakog kupca (grupisano po imenu i prezimenu zajedno) i uštedu ostvarenu na sve popuste u njegovim fakturama (izračunatu preko količine jednog proizvoda). Kao vrijednost treće kolone prikazati vaš broj indeksa. Kolone nazvati “Kupac”, “Usteda” i “Indeks”.
8. Ispisati sve isporuke bez ponavljanja isporuke čije fakture imaju popust i broj mjeseci garancije.
9. Ispisati naziv i cijenu proizvoda čija je cijena veća od prosjeka (zaokruženog na dvije decimale) maksimalnih cijena proizvoda iz svake kategorije.
10. Ispisati naziv i cijenu svih proizvoda čija je cijena manja od svih prosječnih cijena svake kategorije koja nije podkategorija kategorije tog proizvoda.

U datoteci Provjera.sql se nalaze upiti koji vam pomažu da provjerite da li je vaš upit tačan (Obratite pažnju da morate tačno ispoštovati kako se trebaju zvati kolone). Zadatak provjeravate tako što dio ZAMJENA zamjenite sa vašim upitom i pokrenete upit. Kao rezultat upita dobiti ćete broj. Npr ako je dat upit:

```
SELECT SUM(LENGTH(DRZAVA)*3)
FROM (ZAMJENA);
```

A trebate ispisati nazive država onda je potrebno napisati upit koji ako vraća rezultat 183 upit je ispravan:

```
SELECT SUM(LENGTH(DRZAVA)*3)
FROM (SELECT naziv AS Drzava FROM Drzava);
```

Rezultati upita su:

1. 207
2. 402
3. 51
4. 504
5. 6897
6. 18
7. 17709
8. 243
9. 9210
10. 2448

Za slanje na c2 trebate dobiti brojeve na isti način, samo što koristite drugu datoteku pod nazivom SLANJE.sql

Zbog mogućih problema sa brzinom pristupa fakultetskoj bazi podataka u slučaju preopterećenosti, za rad na zadacima 2 i 3 alternativno je moguće koristiti Oracle Live okruženje (https://livesql.oracle.com).

Zadatak 2 (5b)

Zadatak 2 se postavlja u SQL skripti koja se šalje na zamger, odmah ispod zadatka 1 sa komentarom --Zadatak 2. Na slici 1 su date tri tabele, pri čemu su *null* vrijednosti prikazane simbolom "-". Potrebno je kreirati ove tri tabele i popuniti ih odgovarajućim podacima. Uslovi za tabele su da je **FKTabelaA** strani ključ na *id* kolonu tabele *TabelaA* i ne smije biti *null,* a **FKTabelaB** strani ključ na *id* kolonu tabele *TabelaB* i smije biti *null*. Kolona *realniBroj* tabele *TabelaA* mora biti veći od 5, a kolona *cijeliBroj* ne smije biti između 5 i 15. Kolona *cijeliBroj* od tabele *TabelaB* se ne smije ponavljati. Kolone *naziv* i *cijeliBroj* tabele *TabelaC* ne smiju biti *null*. Ograničenje stranog ključa na tabelu *TabelaB* je potrebno nazvati "*FkCnst*".

ID	NAZIV	DATUM	CIJELIBROJ	REALNIBROJ
1	tekst	-	-	6.2
2	-	-	3	5.26
3	tekst	-	1	-
4	-	-	-	-
5	tekst	-	16	6.78

TabelaA

ID	NAZIV	DATUM	CIJELIBROJ	REALNIBROJ	FKTABELAA
1	-	-	1	-	1
2	-	-	3	-	1
3	-	-	6	-	2
4	-	-	11	-	2
5	-	-	22	-	3

TabelaB

ID	NAZIV	DATUM	CIJELIBROJ	REALNIBROJ	FKTABELAB
1	YES	-	33	-	4
2	NO	-	33	-	2
3	NO	-	55	-	1

TabelaC

Slika 1. Izgled tabela baze podataka

Pozivanjem sljedećih 10 INSERT komandi jedne za drugom, utvrditi za svaku komandu da li se može izvršiti ili ne, i zašto. Svaka komanda se poziva nad tabelom koja ima rezultujuće podatke prethodnih komandi koje su se uspješno izvršile.

```
1.  INSERT INTO TabelaA (id, naziv, datum, cijeliBroj, realniBroj) VALUES (6, 'tekst', null, null,
    6.20);
2.  INSERT INTO TabelaB (id, naziv, datum, cijeliBroj, realniBroj, FkTabelaA) VALUES (6, null,
    null, 1, null, 1);
3.  INSERT INTO TabelaB (id, naziv, datum, cijeliBroj, realniBroj, FkTabelaA) VALUES (7, null,
    null, 123, null, 6);
4.  INSERT INTO TabelaC (id, naziv, datum, cijeliBroj, realniBroj, FkTabelaB) VALUES (4, 'NO',
    null, 55, null, null);
5.  UPDATE TabelaA SET naziv = 'tekst' WHERE naziv IS NULL AND cijeliBroj IS NOT NULL;
6.  DROP TABLE TabelaB;
7.  DELETE FROM TabelaA WHERE realniBroj IS NULL;
8.  DELETE FROM TabelaA WHERE id = 5;
9.  UPDATE TabelaB SET fktabelaA = 4 WHERE fktabelaA = 2;
10. ALTER TABLE TabelaA ADD CONSTRAINT cst CHECK (naziv LIKE 'tekst');
```

Rezultati za provjeru su:

```
SELECT SUM(id) FROM TabelaA; --Rezultat: 16
SELECT SUM(id) FROM TabelaB; --Rezultat: 22
SELECT SUM(id) FROM TabelaC; --Rezultat: 10
```

Zadatak 3 (5b)

Zadatak 3 se postavlja u SQL skripti koja se šalje na zamger, odmah ispod zadatka 2 sa komentarom --Zadatak 3. Obrisati sve tabele iz zadatka 2, a zatim ih ponovo kreirati na isti način kao u prvom paragrafu zadatka 2 (kako bi se dobilo isto stanje tabela i ograničenja kao na Slici 1, bez izvršavanja 10 INSERT komandi iz zadatka 2).

Kreirati triger **t1** koji, nakon dodavanja svakog novog reda u tabelu *TabelaB*, odgovarajućem redu tabele *TabelaA* povećava vrijednost njegovog realnog broja za 25% ukoliko je vrijednost cijelog broja novog reda tabele *TabelaB* manja od 50, a u suprotnom smanjuje vrijednost njegovog realnog broja za 25%.

Kreirati triger **t2** koji, prije dodavanja ili izmjene svakog reda u tabeli *TabelaC*, dodaje novi red u tabelu *TabelaB*. Novi red treba da referencira isti red tabele *TabelaA* kao i odgovarajući red tabele *TabelaB* na koji se dati red tabele *TabelaC* referencira (u slučaju izmjene, koristiti novu vrijednost). Za datum novog reda postaviti današnji datum, za cijeli broj uduplanu maksimalnu vrijednost cijelog broja iz tabele *TabelaB*, a za id postaviti sljedeću vrijednost sekvence *brojacB* korištene za početni unos redova 1-5 u tabelu *TabelaB*. Ostale vrijednosti kolona postaviti na *null*. Ukoliko sekvencu *brojacB* niste kreirali prije unosa redova, naknadno je kreirati s odgovarajućim postavkama. Za studente koji koriste SQLTools, potrebno je da sekvenca *brojacB* ima minimalnu vrijednost i započinje od vrijednosti 0, a za studente koji koriste Oracle Live, potrebno je da sekvenca *brojacB* započinje od vrijednosti 1.

Kreirati praznu *backup* tabelu *TabelaABekap*. Tabela *TabelaABekap* je identična tabeli *TabelaA*, samo što ima dodatne kolone *cijeliBrojB* (tipa INTEGER) i *sekvenca* (tipa INTEGER). Kreirati triger **t3** koji puni *backup* tabelu *TabelaABekap*. Nakon dodavanja bilo kojeg novog reda u tabelu *TabelaB*, referencirani red tabele *TabelaA* spašava se u *backup* tabelu. Pritom se u kolonu *cijeliBrojB* upisuje *cijeliBroj* od unesenog reda tabele *TabelaB*. Ako je u tabelu *TabelaABekap* već ranije unesen red s istim primarnim ključem, potrebno je samo povećati *cijeliBrojB* kolonu za vrijednost kolone *cijeliBroj* unesenog reda tabele *TabelaB*. U kolonu *sekvenca* je potrebno unijeti

sljedeću vrijednost sekvence *seq* koja je kreirana prije samog trigera. Ova sekvenca poprima vrijednosti između 1 i 10 u kružnom redoslijedu. Za sve studente potrebno je da sekvenca *seq* ima minimalnu vrijednost i započinje od vrijednosti 1 (ne treba se koristiti vrijednost 0 za ovu sekvencu).

Nakon poziva komandi:

```
INSERT INTO TabelaB (id, naziv, datum, cijeliBroj, realniBroj, FkTabelaA) VALUES (brojacB.nextval, null,
null, 2, null, 1);
INSERT INTO TabelaB (id, naziv, datum, cijeliBroj, realniBroj, FkTabelaA) VALUES (brojacB.nextval, null,
null, 4, null, 2);
INSERT INTO TabelaB (id, naziv, datum, cijeliBroj, realniBroj, FkTabelaA) VALUES (brojacB.nextval, null,
null, 8, null, 1);
INSERT INTO TabelaC (id, naziv, datum, cijeliBroj, realniBroj, FkTabelaB) VALUES (4, 'NO', null, 5, null, 3);
INSERT INTO TabelaC (id, naziv, datum, cijeliBroj, realniBroj, FkTabelaB) VALUES (5, 'YES', null, 7, null,
3);
INSERT INTO TabelaC (id, naziv, datum, cijeliBroj, realniBroj, FkTabelaB) VALUES (6, 'NO', null, 9, null, 2);
UPDATE TabelaC SET cijeliBroj = 10 WHERE id = 2;
DELETE FROM TabelaB WHERE id NOT IN (SELECT FkTabelaB FROM TabelaC);
DELETE FROM TabelaA WHERE id IN (3, 4, 6);
```

Rezultati poziva sljedećih SQL iskaza trebaju biti:

```
SELECT SUM(id*3 + cijeliBrojB*3) FROM TabelaABekap; --Rezultat: 2031
SELECT SUM(id*3 + cijeliBroj*3) FROM TabelaC; --Rezultat: 420
SELECT SUM(MOD(id,10)*3) FROM TabelaB; --Rezultat: 30
SELECT SUM(id + realniBroj)*10 FROM TabelaA; --Rezultat: 264
```

Potrebno je utvrditi rezultate poziva sljedećih SQL iskaza:

1. `SELECT SUM(id*7 + cijeliBrojB*7) FROM TabelaABekap;`
2. `SELECT SUM(id*7 + cijeliBroj*7) FROM TabelaC;`
3. `SELECT SUM(MOD(id,10)*7) FROM TabelaB;`
4. `SELECT SUM(id*5 + realniBroj)*10 FROM TabelaA;`