II DEO – SQLgpt i vežbe sa upitima

Zadatak broj 1 – Svi zaposleni koji su radili za više od 3 klijenta

Kod:

```
SELECT
    Z.zaposleni_id,
    Z.ime,
    COUNT(DISTINCT P.klijent_id) AS broj_klijenata
FROM
    Zaposleni Z
JOIN
    Projekti P ON Z.zaposleni_id = P.zaposleni_id
WHERE
    P.datum_pocetka >= DATE_SUB(CURDATE(), INTERVAL 1 YEAR)
GROUP BY
    Z.zaposleni_id, Z.ime
HAVING
    COUNT(DISTINCT P.klijent_id) > 3;
```

Zadatak broj 2 – Svi studenti koji su dobili ocene veće od 8

Kod:

```
SELECT
    S.student_id,
    S.ime,
    COUNT(DISTINCT I.predmet_id) AS broj_polozenih_predmeta
FROM
    Studenti S
JOIN
    Ispiti I ON S.student_id = I.student_id
JOIN
    Predmeti P ON I.predmet_id = P.predmet_id
WHERE
    I.ocena > 8
GROUP BY
    S.student_id, S.ime
HAVING
    COUNT(DISTINCT I.predmet_id) >= 3;
```

Zadatak broj 3 – Zaposleni koji u odeljenju rade duže od 5 godina

Kod:

```
SELECT
    Z.ime,
    Z.prezime,
    Z.datum_zaposlenja
FROM
    Zaposleni Z
JOIN
    Odeljenja O ON Z.odeljenje_id = O.odeljenje_id
WHERE
    O.ime_odeljenja = 'IT'
    AND Z.datum_zaposlenja <= CURDATE() - INTERVAL 5 YEAR;
```

Zadatak broj 4 – Proizvodi koji nisu naručeni

Kod:

```
SELECT
    P.proizvod_id,
    P.naziv
FROM
    Proizvodi P
LEFT JOIN
    Narudžbine N ON P.proizvod_id = N.proizvod_id
WHERE
    N.proizvod_id IS NULL;
```

Zadatak broj 5 – Klijenti koji su potrošili više od 10.000 dinara

Kod:

```
SELECT
    K.klijent_id,
    K.ime,
    N.narudžbina_id,
    SUM(D.cena * D.količina) AS ukupna_vrednost
FROM
    Klijenti K
JOIN
```

```
    Narudžbine N ON K.klijent_id = N.klijent_id
JOIN
    DetaljiNarudžbine D ON N.narudžbina_id = D.narudžbina_id
WHERE
    N.datum_narudžbine >= CURDATE() - INTERVAL 1 YEAR
GROUP BY
    K.klijent_id, N.narudžbina_id
HAVING
    SUM(D.cena * D.količina) > 10000;
```

Zadatak broj 6 -Pacijenti sa više od 3 posete ustanovi

Kod:

```
SELECT
    P.pacijent_id,
    P.ime,
    COUNT(DISTINCT Po.lekar_id) AS broj_razlicitih_lekara
FROM
    Pacijenti P
JOIN
    Posete Po ON P.pacijent_id = Po.pacijent_id
WHERE
    Po.datum_posete >= CURDATE() - INTERVAL 6 MONTH
GROUP BY
    P.pacijent_id
HAVING
    COUNT(DISTINCT Po.lekar_id) > 3;
```

Zadatak broj 7 – Proizvodi čije recenzije su manje od 3

Kod:

```
SELECT
    P.proizvod_id,
    P.naziv,
    AVG(R.ocena) AS prosecna_ocena
FROM
    Proizvodi P
JOIN
    Recenzije R ON P.proizvod_id = R.proizvod_id
GROUP BY
    P.proizvod_id, P.naziv
HAVING
    AVG(R.ocena) < 3;
```

Zadatak broj 8 – Zaposleni koji su radili za više od 3 klijenta

Kod:

```
SELECT
    Z.zaposleni_id,
    Z.ime,
    COUNT(DISTINCT P.klijent_id) AS broj_razlicitih_klijenata
FROM
    Zaposleni Z
JOIN
    Projekti P ON Z.zaposleni_id = P.zaposleni_id
WHERE
    P.datum_pocetka >= CURDATE() - INTERVAL 1 YEAR
GROUP BY
    Z.zaposleni_id
HAVING
    COUNT(DISTINCT P.klijent_id) > 3;
```

Zadatak broj 9 – Artikli kojima je porasla prodaja u poslednje dve godine

Kod:

```
WITH ProdajaPoKvartalu AS (
    SELECT
        p.artikal_id,
        YEAR(p.datum_prodaje) AS Godina,
        DATEPART(QUARTER, p.datum_prodaje) AS Kvartal,
        SUM(p.kolicina_prodaje) AS UkupnaProdaja
    FROM
        Prodaja p
    WHERE
        p.datum_prodaje >= DATEADD(YEAR, -2, GETDATE())
    GROUP BY
        p.artikal_id,
        YEAR(p.datum_prodaje),
        DATEPART(QUARTER, p.datum_prodaje)
),
RastucaProdaja AS (
    SELECT
        a.*,
        LEAD(a.UkupnaProdaja)
        OVER(PARTITION BY a.artikal_id
          ORDER BY a.Godina, a.Kvartal) AS SledecaProdaja
    FROM
```

```
        ProdajaPoKvartalu a
)
SELECT
    DISTINCT a.artikal_id
FROM
    RastucaProdaja a
WHERE
    a.UkupnaProdaja < a.SledecaProdaja
```

Zadatak broj 10 – Korisnici koji su pazarili više od prosečnih kupovina

Kod:

```
WITH ProsečneTransakcije AS (
    SELECT
        t.korisnik_id,
        AVG(t.iznos_transakcije) AS ProsečnaTransakcija
    FROM
        Transakcije t
    GROUP BY
        t.korisnik_id
)
SELECT
    k.*,
    t.iznos_transakcije,
    t.datum_transakcije
FROM
    Korisnici k
JOIN
    Transakcije t ON k.korisnik_id = t.korisnik_id
JOIN
    ProsečneTransakcije pt ON t.korisnik_id = pt.korisnik_id
WHERE
    t.iznos_transakcije > pt.ProsečnaTransakcija
ORDER BY
    t.korisnik_id, t.datum_transakcije;
```

Zadatak broj 11 – Studenti koji nisu položili ispit

Kod:

```
SELECT
    s.*
FROM
```

```
    Studenti s
WHERE NOT EXISTS (
    SELECT
        1
    FROM
        Ispiti i
    WHERE
        i.student_id = s.student_id
        AND i.predmet_id = @odredjeniPredmetId
-- Zamijenite @odredjeniPredmetId sa ID-om konkretnog
-- predmeta
        AND i.ocena > 5
)
```

Zadatak broj 12- Zaposleni koji nisu radili na projektima u inostranstvu

Kod:

```
SELECT
    z.*
FROM
    Zaposleni z
WHERE NOT EXISTS (
    SELECT
        1
    FROM
        Projekti p
    JOIN
        Lokacije l ON p.lokacija_id = l.lokacija_id
    WHERE
        p.zaposleni_id = z.zaposleni_id
        AND l.u_inostranstvu = 1
-- Pretpostavljamo da `u_inostranstvu` je kolona
-- sa logičkim vrednostima
-- koja označava da li je lokacija u inostranstvu
)
```

Zadatak broj 13 – Kreiranje arhive prodaje

Kod:

```
INSERT INTO ArhivaProdaje
   (id_prodaje, id_artikla, kolicina, datum_prodaje)
SELECT id_prodaje, id_artikla, kolicina, datum_prodaje
```

```
FROM Prodaja
WHERE YEAR(datum_prodaje) = YEAR(CURRENT_DATE) - 1;

Kod:
DELETE FROM Prodaja
WHERE YEAR(datum_prodaje) = YEAR(CURRENT_DATE) - 1;
```

Kod:

```
BEGIN TRANSACTION;

-- Izvršite INSERT INTO...SELECT naredbu

-- Izvršite DELETE naredbu

-- Potvrdite transakciju ako su sve naredbe uspešne
COMMIT;
```

Zadatak broj 14 – Ažuriranje tabele dodavanjem novih zapisa

1. Kod:

```
SELECT Artikli.artikal_id
FROM Artikli
LEFT JOIN Zalihe ON Artikli.artikal_id = Zalihe.artikal_id
WHERE Zalihe.artikal_id IS NULL;
```

2. Kod:

```
INSERT INTO Zalihe (artikal_id, kolicina, ...)
SELECT Artikli.artikal_id, 0 AS kolicina, ...
FROM Artikli
LEFT JOIN Zalihe ON Artikli.artikal_id = Zalihe.artikal_id
WHERE Zalihe.artikal_id IS NULL;
```

Zadatak broj 15 – Bonus bodovi svim studentima sa prosečnim ocenama većim od 9

1. Kod:

```
SELECT student_id, AVG(ocena) AS prosecna_ocena
FROM Ispiti
WHERE ocena >= 9 AND YEAR(datum_ispita) = YEAR(CURRENT_DATE)
```

```
GROUP BY student_id
HAVING AVG(ocena) >= 9;
```

2. Kod:

```
INSERT INTO BonusBodovi (student_id, bodovi)
SELECT student_id, 10 AS bodovi
FROM Ispiti
WHERE ocena >= 9 AND YEAR(datum_ispita) = YEAR(CURRENT_DATE)
GROUP BY student_id
HAVING AVG(ocena) >= 9;
```

Zadatak broj 16 – Zaposleni koji nemaju dodeljen projekat

Kod:

```
SELECT
   Z.ID_Zaposlenog,
   Z.Ime,
   Z.Prezime
FROM Zaposleni AS Y
LEFT JOIN ProjektiZaposlenih AS PZ
ON Z.ID_Zaposlenog = PZ.ID_Zaposlenog
WHERE PZ.ID_Projekta IS NULL;
```

Zadatak broj 17 – Studenti koji nisu podneli nijednu prijavu za stipendiju

Kod:

```
SELECT S.student_id, S.Ime, S.Prezime
FROM Studenti AS S
LEFT JOIN StipendijePrijave AS SP
ON S.student_id = SP.student_id
WHERE SP.prijavljeno_datum IS NULL;
```

Zadatak broj 18 – Rang lista studenata

1. Kod:

```
WITH PBP AS (
  SELECT
```

```
    I.ID_Studenta,
    AVG(I.Ocena) AS Prosek
  FROM
    Ispiti AS I
  WHERE
    I.Ocena >= 6
  GROUP BY
    I.ID_Studenta
)
```

Kod:

```
, SSP AS (
  SELECT DISTINCT
    I.ID_Studenta
  FROM
    Ispiti AS I
  WHERE
    I.Ocena < 6
)
```

2. Kod

```
SELECT
  S.ID_Studenta,
  S.Ime,
  S.Prezime,
  PBP.Prosek,
  CASE
    WHEN SSP.ID_Studenta IS NOT NULL THEN 'Da'
    ELSE 'Ne'
  END AS SmanjenZbogPada
FROM
  Studenti AS S
  JOIN ProsekBezPadova AS PBP
    ON S.ID_Studenta = PBP.ID_Studenta
  LEFT JOIN StudentiSaPadom AS SSP
   ON S.ID_Studenta = SSP.ID_Studenta
ORDER BY
  PBP.Prosek DESC;
```

Zadatak broj 19 – Projekti koji premašuju budžet

Kod:

```
WITH UkupniTroškovi AS (
  SELECT
    ID_Projekta,
    SUM(Iznos) AS UkupanIznos
  FROM
```

```
    TroškoviProjekata
  GROUP BY
    ID_Projekta
)

SELECT
  P.ID_Projekta,
  P.Naziv,
  P.Budžet,
  UkupniTroškovi.UkupanIznos
FROM
  Projekti AS P
  JOIN      UkupniTroškovi      ON      P.ID_Projekta      =
UkupniTroškovi.ID_Projekta
WHERE
  UkupniTroškovi.UkupanIznos > P.Budžet;
```

Kod:

```
SELECT
  Projekti.ID_Projekta,
  Projekti.Naziv,
  Projekti.Budžet,
  (SELECT SUM(Iznos) FROM TroškoviProjekata
WHERE TroškoviProjekata.ID_Projekta = Projekti.ID_Projekta)
AS UkupanIznos
FROM
  Projekti
HAVING
  UkupanIznos > Projekti.Budžet;
```

Zadatak broj 20 – Ažuriranje statusa narudžbina

Kod:

```
UPDATE Narudžbine
SET ID_Statusa = (
   SELECT ID_Statusa FROM StatusNarudžbine
   WHERE OpisStatusa = 'Otkazano'
)
WHERE DatumNarudžbine <= CURDATE() - INTERVAL 30 DAY
AND ID_Statusa != (
   SELECT ID_Statusa FROM StatusNarudžbine
   WHERE OpisStatusa = 'Otkazano'
);
```

Zadatak broj 21 – Ažuriranje statusa zaposlenih na određeni nivo

Kod:

```
UPDATE Zaposleni AS Z
JOIN StatusiZaposlenih AS SZ
ON Z.ID_Statusa = SZ.ID_Statusa
SET Z.ID_Statusa = (
  SELECT ID_Statusa FROM StatusiZaposlenih
  WHERE OpisStatusa = 'Senior'
)
WHERE (SZ.OpisStatusa = 'Junior' OR SZ.OpisStatusa = 'Mid-
level')
AND DATEDIFF(CURDATE(), Z.DatumPočetkaRada) > 365 * 5;
```

1.

Zadatak broj 22 – Ažuriranje kontaktnih informacija klijenata

Kod:

```
UPDATE Klijenti
JOIN
   PrivremeniKontakti
ON
   Klijenti.ID_Klijenta = PrivremeniKontakti.ID_Klijenta
SET
   Klijenti.Email = PrivremeniKontakti.Email,
   Klijenti.BrojTelefona = PrivremeniKontakti.BrojTelefona
WHERE
   PrivremeniKontakti.DatumAžuriranja IS NOT NULL;
```

1.

Zadatak broj 23 – Implementacija logike za automatsko promovisanje zaposlenih

Kod:

```
UPDATE Zaposleni
SET ID_Statusa = (
  SELECT ID_Statusa
  FROM StatusiZaposlenih
  WHERE OpisStatusa = (
    CASE
      WHEN (
```

```
        SELECT OpisStatusa
        FROM StatusiZaposlenih
        WHERE ID_Statusa = Zaposleni.ID_Statusa
      ) = 'Novajlija'
      THEN 'Srednji'
      WHEN (
        SELECT OpisStatusa
        FROM StatusiZaposlenih
        WHERE ID_Statusa = Zaposleni.ID_Statusa
      ) = 'Srednji'
      THEN 'Napredni'
      WHEN (
        SELECT OpisStatusa
        FROM StatusiZaposlenih
        WHERE ID_Statusa = Zaposleni.ID_Statusa
      ) = 'Napredni'
      THEN 'Ekspert'
      ELSE 'Ekspert'
    END
  )
)
WHERE Zaposleni.ID_Zaposlenog IN (
  SELECT P.ID_Zaposlenog
  FROM Performanse P
  JOIN Zaposleni Z ON P.ID_Zaposlenog = Z.ID_Zaposlenog
  WHERE P.Ocena > 8
  AND DATEDIFF(CURDATE(), Z.DatumPočetkaRada) > 365 * 2
);
```

Zadatak broj 24 – Optimizovanje zaliha ažuriranjem minimalnih količina

Kod:

```
UPDATE Zalihe
SET MinimalnaKoličinaZaliha = (
  SELECT
    AVG(Prodaja.KoličinaProdaje) * FaktorSigurnosneZalihe
    AS NovaMinimalnaKoličina
  FROM Prodaja
  WHERE Prodaja.ID_Proizvoda = Zalihe.ID_Proizvoda
  GROUP BY Prodaja.ID_Proizvoda
)
WHERE EXISTS (
  SELECT 1
  FROM Prodaja
  WHERE Prodaja.ID_Proizvoda = Zalihe.ID_Proizvoda
);
```

-

Zadatak broj 25 – Izveštaj o zaposlenima i svim klijentima

Kod:

```
(
  SELECT
    Ime,
    Prezime,
    Departman AS 'Departman/Grad',
    'Zaposleni' AS Tip
  FROM
    Zaposleni
)
UNION ALL
(
  SELECT
    Ime,
    Prezime,
    Grad AS 'Departman/Grad',
    'Klijent' AS Tip
  FROM
    Klijenti
)
ORDER BY
  Prezime,
  Ime;
```

Zadatak broj 26 – Spisak zaposlenih i klijenata kojima je rođendan određenog meseca

Kod:

```
(
  SELECT
    Ime,
    Prezime,
    DatumRođenja,
    'Zaposleni' AS Tip
  FROM
    Zaposleni
  WHERE
    MONTH(DatumRođenja) = MONTH(CURDATE())
)
UNION
(
  SELECT
```

```
    Ime,
    Prezime,
    DatumRođenja,
    'Klijent' AS Tip
  FROM
    Klijenti
  WHERE
    MONTH(DatumRođenja) = MONTH(CURDATE())
)
ORDER BY
  DatumRođenja;
```

Zadatak broj 27 – Dizajniranje strukture tabele Skladište

Kod:

```
CREATE TABLE Skladište (
  ID_Artikla INT AUTO_INCREMENT,
  Naziv VARCHAR(255),
  Kategorija VARCHAR(255),
  KoličinaNaStanju INT,
  PRIMARY KEY (ID_Artikla)
);
```

Zadatak broj 28 – Ažuriranje strukture tabele Zaposleni

Kod:

```
ALTER TABLE Zaposleni
ADD COLUMN zaposleni_id INT AUTO_INCREMENT PRIMARY KEY;
```

-

Zadatak broj 29 – Rangiranje na osnovu određenog kriterijuma

Kod:

```
SELECT
  ID_Zaposlenog,
  Odsek,
  GodišnjiRezultat,
  RANK() OVER (
```

```
    PARTITION BY Odsek
    ORDER BY GodišnjiRezultat DESC
  ) AS Rang
FROM
  PerformanseZaposlenih;
```

Zadatak broj 30 – Identifikovanje najboljih 10% studenata

Kod:

```
WITH ProsečneOcene AS (
  SELECT
    ID_Studenta,
    GodinaStudija,
    AVG(Ocena) AS ProsečnaOcena
  FROM
    Ocene
  GROUP BY
    ID_Studenta,
    GodinaStudija
),
RangiraniStudenti AS (
  SELECT
    *,
    PERCENT_RANK() OVER (
      PARTITION BY GodinaStudija
      ORDER BY ProsečnaOcena DESC
  ) AS ProcentniRang
  FROM
    ProsečneOcene
)
SELECT
  ID_Studenta,
  GodinaStudija,
  ProsečnaOcena,
  'Najbolji student' AS Status
FROM
  RangiraniStudenti
WHERE
  ProcentniRang <= 0.10;
```

-

Zadatak broj 31 – Spisak 5 najskupljih proizvoda unutar kategorije

Kod:

```
WITH RangiraniProizvodi AS (
  SELECT
    ID_Proizvoda,
    Naziv,
    Kategorija,
    Cena,
    ROW_NUMBER() OVER (
      PARTITION BY Kategorija
      ORDER BY Cena DESC
    ) AS Rang
  FROM
    Proizvodi
)
SELECT
  ID_Proizvoda,
  Naziv,
  Kategorija,
  Cena
FROM
  RangiraniProizvodi
WHERE
  Rang <= 5
ORDER BY
  Kategorija,
  Cena DESC;
```

Zadatak broj 32 – Prvih 5 zaposlenih sa najdužim stažom

Kod:

```
SELECT
  ID_Zaposlenog,
  Ime,
  DatumPočetkaRada
FROM
  Zaposleni
ORDER BY
  DatumPočetkaRada
LIMIT 5;
```

-

Zadatak broj 33 – Proizvod sa drugom najvišom cenom

Kod:

```
WITH RangiraniProizvodi AS (
    SELECT
        ID_proizvoda,
        Naziv,
        Kategorija,
        Cena,
        DENSE_RANK() OVER (
          PARTITION BY Kategorija
          ORDER BY Cena DESC
        ) AS Rang
    FROM
        Proizvodi
)
SELECT
    ID_proizvoda,
    Naziv,
    Kategorija,
    Cena
FROM
    RangiraniProizvodi
WHERE
    Rang = 2;
```

Zadatak broj 34 – Selekcija prvih 50% proizvoda unutar kategorije

Kod:

```
WITH RangiraniProizvodi AS (
    SELECT
        ID_proizvoda,
        Naziv,
        Kategorija,
        Cena,
        PERCENT_RANK() OVER (
          PARTITION BY Kategorija
          ORDER BY Cena DESC
        ) AS ProcenatRanga
    FROM
        Proizvodi
)
SELECT
    ID_proizvoda,
```

```
    Naziv,
    Kategorija,
    Cena
FROM
    RangiraniProizvodi
WHERE
    ProcenatRanga <= 0.5;
```

Kod:

```
SELECT
    ID_proizvoda,
    Naziv,
    Kategorija,
    Cena
FROM (
    SELECT
        ID_proizvoda,
        Naziv,
        Kategorija,
        Cena,
        PERCENT_RANK() OVER (
          PARTITION BY Kategorija
          ORDER BY Cena DESC
        ) AS ProcenatRanga
    FROM
        Proizvodi
) AS RangiraniProizvodi
WHERE
    ProcenatRanga <= 0.5;
```

Zadatak broj 35 – Poslednih 25% narudžbina po danu

Kod:

```
SELECT
    ID_narudzbine,
    ID_klijenta,
    datum_narudzbine,
    iznos
FROM (
    SELECT
        ID_narudzbine,
        ID_klijenta,
        datum_narudzbine,
        iznos,
        PERCENT_RANK() OVER (
```

```
          PARTITION BY ID_klijenta
          ORDER BY datum_narudzbine DESC
        ) as Rang
    FROM Narudzbine
) AS RangiraneNarudzbine
WHERE Rang > 0.75;
```