



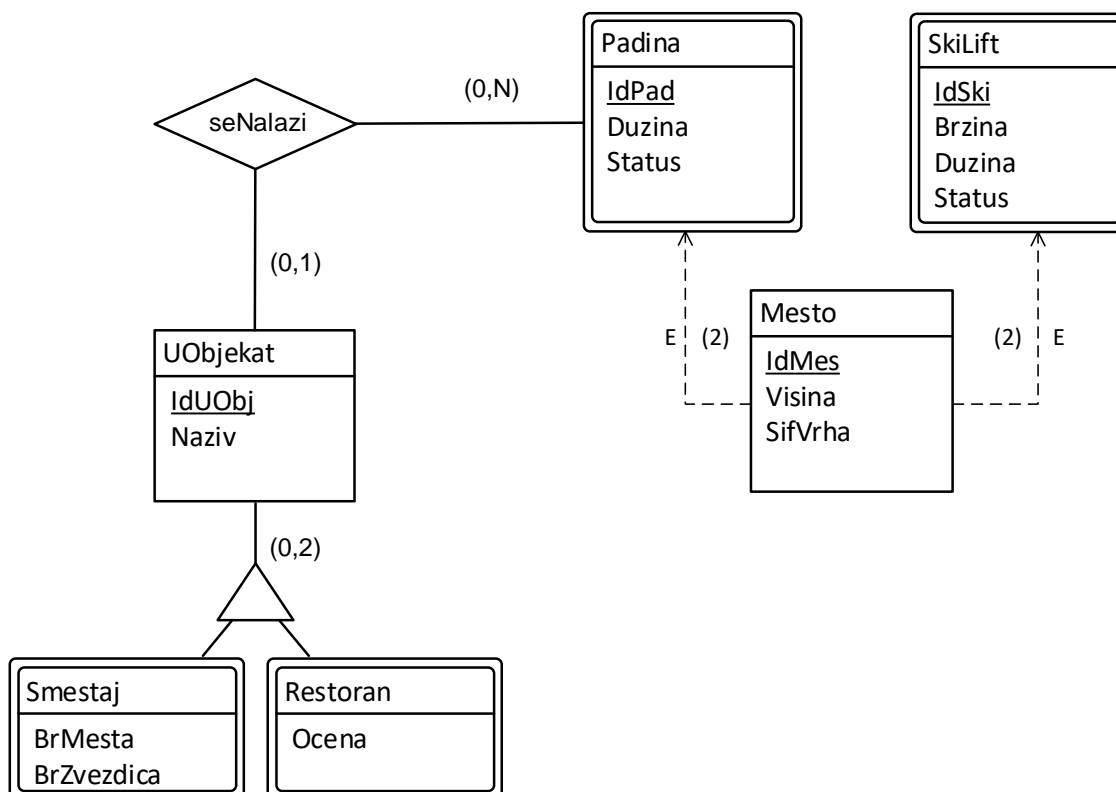
## Базе података 1

(13C112БП1, 13E113БП1, 13E113БП)  
- фебруарски испитни рок -

### Група Б

Посматра се база података једног ски центра који се налази на обронцима планинских врхова. На обронцима планинских врхова се налазе падине, ски лифтови и угоститељски објекти. На обронцима планинских врхова налазе се места од интереса. За сваки планински врх постоји барем једно место од интереса. Места карактеришу висина и шифра планинског врха на чијем обронку се то место налази. Падину карактерише дужина, статус ('r' – ради, 'n' – не ради), почетак и крај падине. Почетак и крај падине представљају места од интереса. Сматрати да је место почетка падине увек више од места краја падине и да оба места припадају истом обронку планинског врха. За ски лифтове се прати брзина, дужина, статус ('r' – ради, 'n' – не ради), почетак и крај ски лифта. Почетак и крај ски лифта представљају места од интереса. Сматрати да је место почетка ски лифта увек ниже од места краја ски лифта и да оба места припадају истом обронку планинског врха. Сматрати да скијаши никада не скијају узбрдо. На почетак сваке падине се може доћи ски лифтом или неком другом падином. У бази се прате угоститељски објекти (назив) од којих неки могу да буду угоститељски објекти са смештајем (број места, број звездица) или ресторани (оцена). Угоститељски објекти се могу (али и не морају) налазити на некој падини. Све дужине и висине су дате у метрима.

У наставку је дата релациона шема посматране базе података.



#### UObjekat (IdUObj, Naziv)

IdUObj	- цео број, идентификује угоститељски објекат, аутоматско додељивање наредног идентификатора
Naziv	- низ до 50 знакова, обавезно

#### Smestaj (IdUObj, BrMesta, BrZvezdica)

IdUObj	- цео број, идентификује угоститељски објекат који је смештај, страни кључ (табела UObjekat)
BrMesta	- цео број, вредност већа од 0
BrZvezdica	- цео број, вредност већа од 0 и мања једнака од 5

#### Restoran (IdUObj, Ocena)

IdUObj	- цео број, идентификује угоститељски објекат који је ресторан, страни кључ (табела UObjekat)
Ocena	- реалан број, вредност већа од 0 и мања једнака од 10

#### Mesto (IdMes, Visina, SifVrha)

IdMes	- цео број, идентификује место, аутоматско додељивање наредног идентификатора
Visina	- цео број, обавезно поље, вредност већа од 0
SifVrha	- низ од тачно 3 знака, обавезно

#### Padina (IdPad, Duzina, Status, IdMesP, IdMesK)

IdPad	- цео број, идентификује падину, аутоматско додељивање наредног идентификатора
Duzina	- цео број, обавезно поље, вредност већа од 0
Status	- тачно 1 знак, обавезно поље, могуће вредности су 'r' (ради) и 'n' (не ради)
IdMesP	- страни кључ (табела Mesto), обавезно
IdMesK	- страни кључ (табела Mesto), обавезно

#### SkiLift (IdSki, Brzina, Duzina, Status, IdMesP, IdMesK)

IdSki	- цео број, идентификује падину, аутоматско додељивање наредног идентификатора
Brzina	- цео број, вредност већа од 0
Duzina	- цео број, обавезно, вредност већа од 0
Status	- тачно 1 знак, обавезно поље, могуће вредности су 'r' (ради) и 'n' (не ради)
IdMesP	- страни кључ (табела Mesto), обавезно
IdMesK	- страни кључ (табела Mesto), обавезно

#### seNalazi (IdUObj, IdPad)

IdUObj	- страни кључ (табела UObjekat), обавезно
IdPad	- страни кључ (табела Padina), обавезно

Задатак 1 [4 поена]

Потребно је направити SQL упит који приказује све угоститељске објекте који су или само смештај или само ресторан. Сортирати резултат по IdUObj опадајуће.

Резултат дати у форми: IdUObj, Naziv

У Cactus-у користити таб: Zadatak 1

---

```
SELECT *
FROM UObjekat
WHERE (
    IdUObj IN (SELECT IdUObj FROM Smestaj)
    AND
    IdUObj NOT IN (SELECT IdUObj FROM Restoran)
) OR (
    IdUObj NOT IN (SELECT IdUObj FROM Smestaj)
    AND
    IdUObj IN (SELECT IdUObj FROM Restoran)
)
ORDER BY IdUObj DESC
```

---

Задатак 2 [4 поена]

Потребно је написати SQL скрипту која убрзава за 10% оне ски лифтове којима би скијаш могао једном падином да се спусти од краја ски лифта до почетка тог ски лифта. Након наведеног ажурирања, потребно је исписати све ски лифтове. Сортирати резултат по IdSki растуће.

Резултат дати у форми: IdSki, Brzina, Duzina

У Cactus-у користити таб: Zadatak 2

---

```
UPDATE SkiLift
SET Brzina=Brzina*1.1
WHERE EXISTS (
    SELECT *
    FROM Padina
    WHERE Padina.IdMesP=SkiLift.IdMesK AND Padina.IdMesK = SkiLift.IdMesP
);

SELECT IdSki, Brzina, Duzina
FROM SkiLift
ORDER BY IdSki
```

---

### Задатак 3 [4 поена]

Потребно је направити SQL упит који исписује за сваки врх који ски лифт (или ски лифтови ако их има више) тренутно пење скијаше до највише тренутне могуће висине на обронку тог врха. Лифтови који не раде не треба да се рачунају. Сматрати да на обронку сваког врха има барем један ски лифт који ради. Сортирати резултат по SifVrha растуће, а затим по IdSki растуће.

Резултат дати у форми: IdSki, SifVrha, Duzina

У Cactus-у користити таб: Zadatak 3

---

```
SELECT IdSki, SifVrha, Duzina
FROM SkiLift S JOIN Mesto M ON(S.IdMesK=M.IdMes)
WHERE S.Status='r' AND NOT EXISTS (
    SELECT *
    FROM SkiLift S2 JOIN Mesto M2 ON(S2.IdMesK=M2.IdMes)
    WHERE M2.SifVrha=M.SifVrha AND M2.Visina>M.Visina AND S2.Status='r'
)
ORDER BY SifVrha, IdSki
```

---

### Задатак 4 [4 поена]

Потребно је написати SQL упит који за сваки врх исписује колико има ски лифтова дужих од 600m на обронцима тог врха. Сортирати резултат по SifVrha опадајуће.

Резултат дати у форми: SifVrha, Broj dugih ski liftova

У Cactus-у користити таб: Zadatak 4

---

```
SELECT SifVrha, COUNT(IdSki) AS "Broj dugih ski liftova"
FROM Mesto LEFT JOIN (SELECT * FROM SkiLift WHERE Duzina>600) ON (IdMes=IdMesK)
GROUP BY SifVrha
ORDER BY SifVrha DESC
```

---

### Задатак 5 [5 поена]

Потребно је направити SQL скрипту која ако постоји табела **Padina** избацује табелу **Padina** из шеме, а затим формира нову табелу **Padina** која треба да има одговарајућу структуру и ограничења.

У Cactus-у користити таб: Zadatak 5

---

```
DROP TABLE IF EXISTS Padina;

CREATE TABLE Padina (
    IdPad          INTEGER PRIMARY KEY AUTOINCREMENT,
    Duzina         INTEGER CHECK(Duzina>0) NOT NULL,
    Status         CHAR CHECK(Status IN ('n','r')) NOT NULL,
    IdMesP         INTEGER NOT NULL REFERENCES Mesto (IdMes),
    IdMesK         INTEGER NOT NULL REFERENCES Mesto (IdMes)
)
```

---

### Задатак 6 [5 поена]

Потребно је написати SQL упит који исписује падину која ради и на којој се налази највише ресторана. Ако постоји више таквих падина исписати их све. Сортирати резултат по Duzini растуће, а затим по IdPad опадајуће.

Резултат дати у форми: IdPad, VisinaOd, VisinaDo, Duzina

У Сactus-у користити таб: Zadatak 6

**Није дозвољено коришћење погледа.**

---

```
WITH PadineURadu ( IdPad, VisinaOd, VisinaDo, Duzina, BR)AS (  
    SELECT IdPad, mp.Visina, mK.Visina, Duzina , COUNT(*)  
    FROM Padina JOIN seNalazi USING(IdPad) JOIN Restoran USING(IdUObj)  
        JOIN Mesto mP ON (IdMesP = mP.IdMes)  
        JOIN Mesto mK ON (IdMesK = mK.IdMes)  
    WHERE Status='r'  
    GROUP BY IdPad  
)  
SELECT IdPad, VisinaOd, VisinaDo, Duzina  
FROM PadineURadu  
WHERE BR= (SELECT MAX(BR) FROM PadineURadu)  
ORDER BY Duzina, IdPad DESC
```

---

### Задатак 7 [5 поена]

Потребно је направити SQL упит који за сваку падину исписује да ли је *blaga*, *srednja* или *strma* у зависности од нагиба падине ( $\alpha$  – нагиб падине). Падина је:

- *blaga* : ако је  $0^\circ \leq \alpha < 45^\circ$
- *srednja* : ако је  $45^\circ \leq \alpha < 70^\circ$
- *strma* : ако је  $70^\circ \leq \alpha \leq 90^\circ$

Сортирати резултат по типу падине растуће (прво стрме, па средње па благе), а затим по IdPad опадајуће.

Препоручена документација: [Built-In Mathematical SQL Functions \(sqlite.org\)](https://www.sqlite.org/lang_expr.html)

Резултат дати у форми: IdPad, Duzina, Tip

У Сactus-у користити таб: Zadatak 7

**Није дозвољено коришћење погледа.**

---

```
WITH PadineNagib AS (  
    SELECT IdPad, Duzina, asin((mP.Visina - mK.Visina)*1.0/Duzina)*180/pi() AS Nagib  
    FROM Padina JOIN Mesto mP ON (Padina.IdMesP = mP.IdMes)  
        JOIN Mesto mK ON (Padina.IdMesK = mK.IdMes)  
)  
SELECT IdPad, Duzina, CASE WHEN Nagib<45 THEN 'blaga'  
    WHEN Nagib<70 THEN 'srednja'  
    ELSE 'strma' END AS Tip  
FROM PadineNagib  
ORDER BY Tip DESC, IdPad DESC
```

---

### Задатак 8 [6 поена]

Потребно је направити SQL упит који проналази три најпопуларнија места ски центра. Рачунање популарности се рачуна кроз сумирање поена које се добијају ако је задовољен услов. Место са више поена популарности је популарније место.

Место добија:

- 2 поена популарности за сваки ски лифт који почиње или се завршава у том месту
- 0.5 поена популарности за сваку падину која почиње или се завршава у том месту
- 0.5 поена популарности за сваки угоститељски објект који није ни смештај ни ресторан који се налази на падини која се завршава у том месту
- 1 поен популарности за сваки ресторан који није смештај и који се налази на падини која се завршава у том месту
- 1.5 поена популарности за сваки смештај који није ресторан и који се налази на падини која се завршава у том месту
- 2 поена популарности за сваки угоститељски објект који је и смештај и ресторан и који се налази на падини која се завршава у том месту

Уколико по поенима популарности има више места коју упадају у три најпопуларнија места приказати само она три са нижим IdMes. Сортирати резултат по PoeniPopularnosti опадајуће, а затим по IdMes растуће.

Резултат дати у форми: IdMes, PoeniPopularnosti

У Cactus-у користити таб: Zadatak 8

**Није дозвољено коришћење погледа.**

---

```
SELECT IdMes,
       2*(SELECT COUNT(*) FROM SkiLift WHERE IdMesP=IdMes OR IdMesK=IdMes)+
       0.5*(SELECT COUNT(*) FROM Padina WHERE IdMesK=IdMes OR IdMesP=IdMes)+
       0.5*(SELECT COUNT(*) FROM seNalazi JOIN Padina USING (IdPad)
           WHERE IdMesK=IdMes)+
       0.5*(SELECT COUNT(*) FROM Restoran JOIN seNalazi USING(IdUObj)
           JOIN Padina USING (IdPad) WHERE IdMesK=IdMes)+
       (SELECT COUNT(*) FROM Smestaj JOIN seNalazi USING(IdUObj)
           JOIN Padina USING (IdPad) WHERE IdMesK=IdMes)
       AS PoeniPopularnosti
FROM Mesto
ORDER BY PoeniPopularnosti DESC, IdMes
LIMIT 3
```

---

Задатак 9 [6 поена]

Скијаш се налази на највишем месту ски центра (сматрати да постоји само једно такво место) и интересује га ако би само скијао до почетка ски лифта у колико ресторана на том путу би могао да се одмори. Обратити пажњу да до почетка ски лифта потенцијално може да се дође различитим путевима (падинама), онда треба изабрати пут са највећим бројем ресторана. Ако скијаш дође до неке падине која не ради онда не може том падином доћи до неке наредне. Потребно је написати SQL упит којим би се пребројало у колико ресторана на путу до сваког ски лифта скијаш може да се одмори. Уколико скијаш не може да дође до почетка неког ски лифта без коришћења неког ски лифта, потребно је исписати -1 за број ресторана. Сортирати резултат по БројRestorana опадајуће, а затим по IdSki растуће.

Резултат дати у форми: IdSki, BrojRestorana

У Sactus-у користити таб: Zadatak 9

**Није дозвољено коришћење погледа.**

---

```
WITH RECURSIVE Spustanje (IdMes, BrojRestorana) AS (  
    SELECT IdMes, 0  
    FROM Mesto  
    WHERE Visina = (SELECT MAX(Visina) FROM Mesto)  
    UNION  
    SELECT IdMesK, BrojRestorana + (  
        SELECT COUNT(*)  
        FROM Restoran NATURAL JOIN seNalazi  
        WHERE seNalazi.IdPad=Padina.IdPad  
    )  
    FROM Spustanje JOIN Padina ON ( IdMesP=IdMes)  
    WHERE Status='r'  
)  
SELECT IdSki, (  
    SELECT COALESCE(MAX(BrojRestorana),-1)  
    FROM Spustanje  
    WHERE IdMes=SkiLift.IdMesP  
    ) AS BrojRestorana  
FROM SkiLift  
ORDER BY BrojRestorana DESC, IdSki
```

---

Задатак 10 [7 поена]

Скијаш се налази у месту са идентификатором 1 и жели да дође до највишег места ски центра. Скијаш је купио 5 карти за ски лифт (приликом сваког коришћења ски лифта потроши једну карту). Скијаш може да скија падинама које раде како би дошао до неког другог ски лифта и тако да се креће по ски центру. Потребно је приказати на које све начине скијаш може да испуни своју жељу, тј. којим падинама и ски лифтовима треба да се креће. Скијаш може више пута на свом путу да користи исти ски лифт, или да скија на истим падинама. У резултату треба приказати колону Putanja која се формира на основу коришћења ски лифтова и падина. Нпр. путања "S1-P2-P5-S3", значи да је скијаш прво узео ски лифт са идентификатором 1 (S1), па затим се спустио падином са идентификатором 2 (P2), па затим се спустио падином са идентификатором 5 (P5), па затим се попео ски лифтом са идентификатором 3 (S3). Ако није могуће испунити жељу скијашу онда треба исписати за путању "Ne postoji put". Сортирати резултат по Putanja опадајуће.

Резултат дати у форми: Putanja

У Cactus-у користити таб: Zadatak 10

**Није дозвољено коришћење погледа.**

---

```
WITH RECURSIVE Korak(IdMesP,IdMesK,Sifra,Tip) AS (  
    SELECT IdMesP, IdMesK, 'P'||IdPad, 0 FROM Padina WHERE Status='r'  
    UNION  
    SELECT IdMesP, IdMesK, 'S'||IdSKi, 1 FROM SkiLift WHERE Status='r'  
),  
Skijanje(IdMes, BrojZica, Putanja) AS (  
    SELECT IdMesK, Tip, Sifra FROM Korak WHERE IdMesP=1  
    UNION  
    SELECT IdMesK, BrojZica+Tip, Putanja||'-'||Sifra  
    FROM Skijanje JOIN Korak ON(IdMes=IdMesP)  
    WHERE BrojZica<5  
),  
SkijanjeDoVrha(Putanja) AS (  
    SELECT Putanja  
    FROM Skijanje JOIN Mesto USING(IdMes)  
    WHERE Visina=(SELECT MAX(Visina) FROM Mesto)  
)  
SELECT * FROM SkijanjeDoVrha  
UNION  
SELECT 'Ne postoji put'  
WHERE NOT EXISTS (SELECT * FROM SkijanjeDoVrha)  
ORDER BY Putanja DESC
```

---