Универзитет у Београду **12.01.2019.**
Електротехнички факултет
Катедра за рачунарску технику и информатику

Базе података 1

(13С112БП1)

- јануарски испитни рок –

Група Б

Посматра се део базе података једне туристичке агенције. У овој бази се прате аранжмани, путници, хотели, места и превоз. Аранжман карактерише назив, цена, датум поласка и доласка и место поласка. Путника карактеришу име, презиме и пол ('M' – мушки, 'Z' – женски). Путник може да уплаћује аранжман, где се прати датум уплате. У систему се прате хотели који се могу наћи у понуди аранжмана. Аранжман има свој план који се састоји из листе места које обухвата. Тим планом је могуће у неком месту одсести у неком од хотела. Хотел карактерише назив хотела, број звездица (1 до 5), капацитет, цена ноћења. Места су дефинисана по називу држави и континенту на коме се налази. Туристичкој агенцији пристижу понуде за превоз које се прате и на основу којих се формирају аранжамни.

У наставку је дата релациона шема посматраног дела базе података.

Putnik (IdPut, ImeIPrezime)

IdPut	- цео број, идентификује путника
ImeIPrezime	- низ до 50 знакова, обавезно
Pol	- један знак, обавезно поље, могуће вредности су 'M' и 'Z' ('M' – мушки, 'Z' – женски)

Aranzman (IdAra, Naziv, DatumOd, DatumDo, Cena)

IdAra	- цео број, идентификује аранжман
Naziv	- низ до 50 знакова, обавезно
DatumOd	- цео број, обавезно поље, датум у формату (yyyymmdd)
DatumDo	- цео број, обавезно поље, датум у формату (yyyymmdd)
Cena	- цео број, обавезно поље, вредности већа од 0

Uplatio (IdPut, IdAra, Datum)

IdPut	- страни кључ (табела Putnik), обавезно
IdAra	- страни кључ (табела Aranzman), обавезно
Datum	- цео број, обавезно поље, датум у формату (yyyymmdd)

Напомена: Комбинација IdPut и IdAra је јединствена.

Drzava (IdDrz, Ime, Kontinent)

IdDrz	- цео број, идентификује државу
Ime	- низ до 50 знакова, обавезно, јединствено
Kontinent	- цео број, обавезно поље, могуће вредности у опсегу 1 до 7

Напомена: Значење атрибута Kontinent: 1-Азија, 2-Антартик, 3-Аустралија, 4-Африка, 5-Европа, 6-Јужна Америка, 7-Северна Америка

Mesto (IdMes, Naziv, IdDrz)

IdMes	- цео број, идентификује место
Naziv	- низ до 50 знакова, обавезно, јединствено
IdDrz	- страни кључ (табела Drzava), обавезно

Hotel (IdHot, Naziv, BrojZvezdica, Kapacitet, Cena, IdMes)

IdHot	- цео број, идентификује хотел, аутоматско додељивање наредног идентификатора
Naziv	- низ до 50 знакова, обавезно
BrojZvezdica	- цео број, обавезно поље, могуће вредности у опсегу 1 до 5
Kapacitet	- цео број, обавезно поље, вредност већа од 0
Cena	- цео број, обавезно поље, вредност већа од 0
IdMes	- страни кључ (табела Mesto), обавезно

PlanAranzmana (IdAra, IdMes, RB, IdHot, BrojNocenja)

IdAra	- страни кључ (табела Aranzman), обавезно
IdMes	- страни кључ (табела Mesto), обавезно
RB	- цео број, обавезно
IdHot	- страни кључ (табела Hotel), није обавезно
BrojNocenja	- цео број, није обавезно поље, вредност већа од 0

Напомена: RB представља редни број по коме се обилази место у једном аранжману. RB се додељује редом почевши од 1. Комбинација IdAra, IdMes и RB је јединствена.

Ако је IdHot недефинисан (NULL), онда се у том месту не ноћи.

Сматрати да је на нивоу базе обезбеђено да се IdMes хотела поклапа са IdMes плана аранжмана.

Prevoz (IdMesOd, IdMesDo, Sredstvo, Cena, Razdaljina)

IdMesOd	- страни кључ (табела Mesto), обавезно
IdMesDo	- страни кључ (табела Mesto), обавезно
Sredstvo	- низ до 7 знакова, обавезно поље, могуће вредности су 'Autobus', 'Avion' и 'Brod'
Cena	- цео број, обавезно поље, вредност већа од 0
Razdaljina	- цео број, обавезно поље, вредност већа од 0

Напомена: Комбинација IdMesOd, IdMesDo и Sredstvo је јединствена.

Задатак 1 [3 поена]

Потребно је направити SQL упит који исписује све новогодишње аранжмане. Новогодишњи аранжмани су аранжмани који у називу садржи реч “nova” и при том је повратак у јануару 2019. Сматрати да наведена реч на почетку назива аранжмана почиње великом словом и да након наведене речи може да буде или бланко знак или зарез или тачка.

Резултат дати у форми: IdAra, Naziv, DatumOd, DatumDo, Cena
У Cactus-у користити таб: Zadatak 1

```
SELECT *
FROM Aranzman
WHERE DatumDo BETWEEN 20190101 AND 20190131 AND
        ( Naziv LIKE '% nova %' OR Naziv LIKE '% nova,%' OR Naziv LIKE '% nova.%'
        OR Naziv LIKE 'nova %' OR Naziv LIKE 'nova,%' OR Naziv LIKE 'nova.%');
```

Задатак 2 [3 поена]

Потребно је направити SQL упит који исписује све датуме аранжмана (без понављања) који крећу из Северне Америке или Јужне Америке. Резултат треба сортирати растуће.

Резултат дати у форми: DatumOd
У Cactus-у користити таб: Zadatak 2

```
SELECT DISTINCT DatumOd
FROM Aranzman NATURAL JOIN PlanAranzmana JOIN Mesto USING(IdMes) NATURAL JOIN Drzava
WHERE (Kontinent=6 OR Kontinent=7) AND PlanAranzmana.RB=1
ORDER BY DatumOd ASC;
```

Задатак 3 [3 поена]

Потребно је направити SQL упит који брише све податке из табеле ***Prevoz*** где се користи *Avion* и где су почетно и крајње место у различитим државама.

Након обрисаних података, исписати све податке из табеле ***Prevoz.***

Резултат дати у форми: IdPre, Sredstvo, Distanca, IdMesOd, IdMesDo
У Cactus-у користити таб: Zadatak 3

```
DELETE FROM Prevoz
WHERE Sredstvo='Avion'  AND EXISTS(
        SELECT *
        FROM Mesto m1, Mesto m2
        WHERE m1.IdMes = Prevoz.IdMesOd AND m2.IdMes = Prevoz.IdMesDo
                AND m1.IdDrz != m2.IdDrz
);

SELECT * FROM Prevoz;
```

Задатак 4 [3 поена]

Потребно је направити SQL упит који исписује за свако место у Јужној Америци колико има хотела са 1, 2 и 3 звездица.

Резултат дати у форми: IdMes,Naziv, 1 Zvezdica, 2 Zvezdice, 3 Zvezdice
У Cactus-у користити таб: Zadatak 4

```
SELECT IdMes, Naziv,
        (SELECT COUNT(*) FROM Hotel WHERE BrojZvezdica=1 AND Hotel.IdMes=m.IdMes)
                AS "1 Zvezdica",
        (SELECT COUNT(*) FROM Hotel WHERE BrojZvezdica=2 AND Hotel.IdMes=m.IdMes)
                AS "2 Zvezdice",
        (SELECT COUNT(*) FROM Hotel WHERE BrojZvezdica=3 AND Hotel.IdMes=m.IdMes)
                AS "3 Zvezdice"
FROM Mesto m NATURAL JOIN Drzava
WHERE Kontinent = 6;
```

Задатак 5 [4 поена]

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

У Cactus-у користити таб Zadatak 5.

```
DROP TABLE IF EXISTS Hotel;

CREATE TABLE Hotel (
        IdHot           INTEGER PRIMARY KEY AUTOINCREMENT,
        Naziv           VARCHAR(50) NOT NULL,
        BrojZvezdica    INTEGER NOT NULL CHECK (BrojZvezdica BETWEEN 1 AND 5),
        Kapacitet       INTEGER NOT NULL CHECK (Kapacitet > 0),
        Cena            INTEGER NOT NULL CHECK (Cena > 0),
        IdMes           INTEGER NOT NULL,
        FOREIGN KEY(IdMes) REFERENCES Mesto(IdMes)
);
```

Задатак 6 [4 поена]

Потребно је направити SQL упит који исписује хотеле заједно са местом, државом и пуним називом континента на коме се налази. Сортирати по IdHot растуће.

Резултат дати у форми: IdHot, Naziv, BrojZvezdica, Mesto, Drzava, Kontinent

У Cactus-у користити таб: Zadatak 6
Није дозвољено коришћење погледа.

```
SELECT IdHot, Hotel.Naziv, BrojZvezdica, Mesto.Naziv AS Mesto, Drzava.Ime AS Drzava,
        CASE Kontinent
                WHEN 1 THEN 'Azija'
                WHEN 2 THEN 'Antartik'
                WHEN 3 THEN 'Australija'
                WHEN 4 THEN 'Afrika'
                WHEN 5 THEN 'Evropa'
                WHEN 6 THEN 'Juzna Amerika'
                ELSE 'Severna Amerika'
        END AS Kontinent
FROM Hotel JOIN Mesto USING (IdMes) NATURAL JOIN Drzava
ORDER BY IdHot ASC;
```

Задатак 7 [4 поена]

Потребно је направити SQL упит који исписује аранжмане у оквиру којег се преноћи (барем једном) у хотилима са барем две различите категорије (различит број звездица). Сортирати по IdAra растуће.

Резултат дати у форми: IdAra, Naziv

У Cactus-у користити таб: Zadatak 7
Није дозвољено коришћење погледа.

```
SELECT IdAra, Naziv
FROM Aranzman WHERE IdAra IN (
        SELECT IdAra
        FROM PlanAranzmana NATURAL JOIN Hotel
        GROUP BY IdAra
        HAVING COUNT(DISTINCT BrojZvezdica) >= 2
)
ORDER BY IdAra;
```

Задатак 8 [4 поена]

Потребно је направити SQL упит којим се увећава цена за 7.5% оних аранжмана чији је укупан број ноћења већи од просечног броја ноћења по аранжману (рачунајући само оне аранжмане којима је предвиђено ноћење).

Након ажурираних података, исписати све податке из табеле ***Aranzman.***
Резултат дати у форми: IdAra, Naziv, DatumOd, DatumDo, Cena

У Cactus-у користити таб: Zadatak 8
Није дозвољено коришћење погледа.

```
UPDATE Aranzman
SET Cena = Cena * 1.075
WHERE IdAra IN (
        SELECT IdAra
        FROM PlanAranzmana
        WHERE IdHot IS NOT NULL
        GROUP BY IdAra
        HAVING SUM(BrojNocenja) > (
                SELECT SUM(BrojNocenja) * 1.0 / COUNT( DISTINCT IdAra)
                FROM PlanAranzmana
                WHERE IdHot IS NOT NULL
        )
);

SELECT * FROM Aranzman;
```

Задатак 9 [5 поена]

Потребно је направити SQL упит којим се формира поглед ***MinimalniKapPoAranzmanuSaSpavanjem (IdAra, Kapacitet)*** који враћа минимални капацитет хотела само за оне аранжмане у којима је предвиђено ноћење у хотелу, а затим је потребно исписати садржај претходно формираног погледа.

У Cactusu-у користити таб Zadatak 9.1 за прављење погледа ***MinimalniKapPoAranzmanuSaSpavanjem***, користити таб Zadatak 9.2 за приказ садржаја ***MinimalniKapPoAranzmanuSaSpavanjem***.

Потребно је направити SQL упит који исписује колико још путника може да уплати аранжман ако је једини ограничавајући фактор минимални капацитет хотела у том аранжману.

У Cactusu-у користити таб Zadatak 9.3 за претходно описан захтев
Резултат дати у форми: IdAra, PreostaloMesta

```
CREATE VIEW MinimalniKapPoAranzmanuSaSpavanjem (IdAra, Kapacitet) AS
SELECT IdAra, MIN(Kapacitet)
FROM PlanAranzmana JOIN Hotel USING (IdHot)
GROUP BY IdAra;

SELECT * FROM MinimalniKapPoAranzmanuSaSpavanjem;

SELECT MinimalniKapPoAranzmanuSaSpavanjem.IdAra, Kapacitet-COUNT(IdPut) AS PreostaloMesta
FROM MinimalniKapPoAranzmanuSaSpavanjem LEFT NATURAL OUTER JOIN Uplatio
GROUP BY IdAra, Kapacitet;
```

Задатак 10 [5 поена]

Потребно је направити SQL упит који исписује имена места и преосталу количину новца које је могуће посетити кренувши из *Beograd*-а са 300 EUR. За превозно средство се користи искључиво аутобус. Ако је могуће на више начина доћи до неког места, потребно је исписати онај резултат код којег остаје већа количина новца. Резултат сортирати опадајуће по преосталој количини новца, а затим по називу места. У резултату не треба да се нађе *Beograd*.

Резултат дати у форми: IdMes, Naziv, Preostalo
У Cactus-у користити таб: Zadatak 10
Није дозвољено коришћење погледа.

```
WITH RECURSIVE Putanje(Id, Novac) AS
(
        SELECT (SELECT IdMes FROM Mesto WHERE Naziv='Beograd'), 300
        UNION
        SELECT Pr.IdMesDo, (Pu.Novac-Pr.Cena)
        FROM Prevoz Pr JOIN Putanje Pu ON (Pr.IdMesOd = Pu.Id)
        WHERE Pr.Sredstvo='Autobus' AND (Pu.Novac-Pr.Cena)>=0
)
SELECT P.Id AS IdMes, M.Naziv, MAX(P.Novac) AS Preostalo
FROM Putanje P JOIN Mesto M ON P.Id = M.IdMes
WHERE M.Naziv != 'Beograd'
GROUP BY P.Id, M.Naziv
ORDER BY 3 DESC, 2 DESC;
```

Задатак 11 [6 поена]

Потребно је направити SQL упит који треба да прикаже којим превозним средством је могуће доћи у *Bec*, ако се полази из *Beograd*-а и притом се у сваком граду који се на путу посети наизменично промени превозно средство: Autobus → Avion → Brod → Autobus → … . У случају да није могуће на овај начин доћи до *Bec*-а, резултат треба да садржи један податак у коме пише "*Nemoguce*".
Ако је могуће доћи до Bec-а, онда је потребно у сваком реду исписати последње коришћено превозно средство (без понављања).

Резултат дати у форми: Sredstvo
У Cactus-у користити таб: Zadatak 11

```
WITH RECURSIVE Doseznost (IdMes, PSredstvo) AS
(
	SELECT IdMes, 2 FROM Mesto WHERE Naziv='Beograd'
	UNION
	SELECT IdMesDo, (PSredstvo + 1)%3
	FROM Prevoz, Doseznost
	WHERE IdMesOd= Doseznost.IdMes AND Sredstvo= (CASE (PSredstvo + 1) % 3
		WHEN 0 THEN 'Autobus'
		WHEN 1 THEN 'Avion'
		ELSE 'Brod' END
	)
)
SELECT DISTINCT CASE PSredstvo
	WHEN 0 THEN 'Autobus'
	WHEN 1 THEN 'Avion'
	ELSE 'Brod' END AS Sredstvo
FROM Doseznost
WHERE IdMes = (SELECT MAX(IdMes) FROM Mesto WHERE Naziv='Bec')
UNION
SELECT 'Nemoguce' AS Sredstvo
WHERE NOT EXISTS(
	SELECT *
	FROM Doseznost
	WHERE IdMes = (SELECT MAX(IdMes) FROM Mesto WHERE Naziv='Bec')
);
```

Задатак 12 [6 поена]

Потребно је направити SQL упит који треба да израчуна цену потенцијалног аранжмана (аранжман још није унет у базу). Цена се формира тако што се на све трошкове аранжмана (превоз и цена хотела) дода 10%. Аранжман би требало да обухвата релацију *Beograd-Budmpesta-Krakov-Beograd*. *Budimpesta* се само обилази – не одседа се у хотелу, док се у *Krakov*-у спава 10 дана у најјефтинијем хотелу који има више од 3 звездице. За цену превоза се тражи најјефтинија понуда, тако да се на релацији Beograd-Budimpesta-Krakov користи аутобус, а на релацији Krakov-Beograd користи авион. Претпоставити да се у бази сигурно налазе одговарајући подаци.

Резултат дати у форми: Cena
У Cactus-у користити таб: Zadatak 12

```
WITH
        Beograd (IdMes) AS (
                SELECT MAX(IdMes) FROM Mesto WHERE Naziv= 'Beograd'),
        Budimpesta (IdMes) AS (
                SELECT MAX(IdMes) FROM Mesto WHERE Naziv= 'Budimpesta'),
        Krakov (IdMes) AS (
                SELECT MAX(IdMes) FROM Mesto WHERE Naziv= 'Krakov')
SELECT SUM(Cena) * 1.10 AS Cena
FROM (
        SELECT 10 * MIN(Cena) AS Cena
        FROM Hotel NATURAL JOIN Krakov
        WHERE BrojZvezdica>3
        UNION ALL
        SELECT MIN(Cena)
        FROM Prevoz, Beograd, Budimpesta
        WHERE Sredstvo='Autobus' AND IdMesOd =Beograd.IdMes AND IdMesDo =Budimpesta.IdMes
        UNION ALL
        SELECT MIN(Cena)
        FROM Prevoz, Budimpesta, Krakov
        WHERE Sredstvo='Autobus' AND IdMesOd = Budimpesta.IdMes AND IdMesDo = Krakov.IdMes
        UNION ALL
        SELECT MIN(Cena)
        FROM Prevoz, Krakov, Beograd
        WHERE Sredstvo='Avion' AND IdMesOd = Krakov.IdMes AND IdMesDo = Beograd.IdMes
) X;
```
