Базе података 1

– надокнада другог колоквијума –

Број индекса (гггг/бббб), име и презиме	Потпис дежурног	Број поена	

Напомена: Није дозвољена употреба литературе. Колоквијум траје **90** минута.

1. (5)Дат је садржај дела базе података о летовима и картама за летове. У табелама *Let* и *Karta* се прате подаци о прошлим, садашњим и будућим летовима као и подаци о картама за те летове.

Let					
IdLet	MestoOd	MestoDo	Poletanje	Status	PotrGoriva
1	Beograd	Rim	20190125	T	0
2	Rim	Pariz	20190127	K	0
3	Beograd	Berlin	20180305	O	0
4	Beograd	NYC	NULL	K	0
5	Rim	Istanbul	20190128	K	0
6	Istanbul	NYC	20180201	Z	5000

Karta			
IdKar	IdLet	Cena	Status
1	1	200	K
2	1	300	N
3	1	100	K
4	2	400	K
5	2	500	K
6	3	700	K

а) Написати *SQL* исказ за прављење табеле *Let*. *IdLet* је целобројна величина која идентификује лет, *MestoOd* и *MestoDo* су низови до 45 карактера и обавезни су, *Poletanje* је целобројна величина, *Status* је карактер ('T' – у току; 'K' - креиран; 'Z' - завршен; 'O' - отказан) и обавезан је, *PotrGoriva* је ненегативан цео број, обавезан је и подразумевана вредност је 0. Остала ограничења треба игнорисати.

```
CREATE TABLE Let ( IdLet INTEGER PRIMARY KEY,
                   MestoOd VARCHAR(45) NOT NULL,
                   MestoDo VARCHAR(45) NOT NULL,
                   Poletanje INT,
                   Status CHAR(1) NOT NULL CHECK(Status IN ('T', 'K', 'Z', 'O')),
                   PotrGoriva INT DEFAULT 0 NOT NULL CHECK(PotrGoriva > 0))
```

б) Одлучено је да се укину сви летови из места „Rim“. Написати *SQL* упит који реализује брисање описаних летова.

```
DELETE FROM Let
WHERE MestoOd = 'Rim'
```

в) Дефинисана је максимална цена 450 за карту. Написати *SQL* упит који регулише цену некупљених карата тако да цене веће од максималне поставља на максималну.

```
UPDATE Karta
SET Cena  = 450
WHERE Cena > 450 AND Status = ‘N’
```

г) Одлучено је да се у продају пусте још две карте за лет 3. Прва карта има *IdKar* 7, а друга 8. Цена прве је 450, друге 400. Написати један *SQL* исказ који ће креирати обе карте.

```
INSERT INTO Karta
VALUES (7, 3, 450, ‘K’), (8, 3, 400, ‘K’)
```

д) Написати *SQL* упит за приказ летова којима је позната информација о полетању (није *NULL* вредност). Приказ резултата треба да буде у формату: *IdLet, Mesto poletanja, Mesto sletanja* (називи колона треба да имају више речи).

```
SELECT  IdLet, MestoOd AS ‘Mesto poletanja', MestoDo AS 'Mesto sletanja'
FROM Let
WHERE Poletanje IS NOT NULL
```

ђ) Написати *SQL* упит за приказ укупне и просечне количине потрошеног горива на завршеним летовима из места *Istanbul*. Приказ треба да буде у формату *Ukupno, Prosek* и сортиран опадајуће по *Ukupno* и опадајуће по *Prosek.*

```
SELECT  SUM(PotrGoriva) AS Ukupno, AVG(PotrGoriva) AS Prosek
FROM Let
WHERE MestoOd = 'Istanbul' AND Status = 'Z'
ORDER BY Ukupno DESC, Prosek DESC
```

е) Потребно је написати *SQL* упит који за сваки лет који није отказан исписује суму зарађеног новца од продаје карата. Приказ резултата треба да буде у формату: *IdLet, MestoOd, MestoDo, Zaradjeno.*

```
SELECT L.IdLet AS IdLet, L.MestoOd AS MestoOd,
L.MestoDo AS MestoDo, SUM(Cena) AS Zaradjeno
FROM Let L, Karta K
WHERE L.IdLet = K.IdLet AND L.Status != ‘O’ AND K.Status = ‘K’
GROUP BY L.IdLet, L.MestoOd, L.MestoDo
```

ж) Потребно је написати *SQL* скрипту која прави поглед (*VIEW*) *KupljeneKarteZaJanuar* који као приказ даје оне карте које су купљене за летове чије је полетање у јануару 2019. Искористити поглед *KupljeneKarteZaJanuar* за приказ просечне цене скупих карата у јануару 2019. Карта је скупа ако јој је цена већа од 500. Приказ резултата треба да буде у формату: *Prosek*.

```
CREATE VIEW KupljeneKarteZaJanuar AS
SELECT K.*
FROM Karta K, Let L
WHERE L.Poletanje >= 20190101 AND L.Poletanje < 20190201
AND K.IdLet = L.IdLet AND K.Status = ‘K’;

SELECT AVG(Cena) AS Prosek
FROM KupljeneKarteZaJanuar
WHERE Cena > 500;
```

з) Потребно је написати *SQL* упит који за сваки град посебно, који је од карата на долазним неотказаним летовима у првом кварталу 2018. године зарадио више од 50000 од продаје карата и продао више од 100 карата, исписује назив тог града, минималну и максималну цену продате карте у формату *MestoDo, MinCena, MaxCena*.

```
SELECT L.MestoDo AS MestoDo, MIN(K.Cena) AS MinCena, Max(K.Cena) AS MaxCena
FROM Let L, Karta K
WHERE L.IdLet = K.IdLet AND L.Status != ‘O’ AND L.Poletanje >= 20180101 AND L.Poletanje <
20180501 AND K.Status = ‘K’
GROUP BY L.MestoDo
HAVING (SUM(K.Cena) > 50000 AND COUNT(*) > 100);
```

2.(10) Дати су шема релације R(S, T, E, F, A, N) и скуп функцијских зависности F={SN→A, FT→N, TS→EA, EF→ST}. Потребно је:

а) Одредити скуп кандидат кључева KK дате шеме.

Одговор:

KK={FE, FTS}

б) Испитати редом да ли је шема у BC, 3. і 2. нормалној форми и сваки пут у табели назначити да ли посматрана зависност нарушава посматрану нормалну форму

	SN→A	FT→N	TS→EA	EF→ST
BCNF	x	x	x	√
3NF	x	x	x	√
2NF	√	x	x	√

в) Спровести нормализацију дате шеме у 3. нормалну форму алгоритмом који гарантује чување функцијских зависности.

Одговор:

R_1(S, N, A) R_2(F, T, N) R_3(T, S, E, A) R_4(E, F, S, T)

г) Спровести нормализацију дате шеме у BC нормалну форму, издвајајући зависности редоследом слева на десно.

Одговор:

R_1(S, N, A) R_2(F, T, N) R_3(S, T, E) R_4(S, T, F)

д) Испитати да ли је при поступку у оквиру тачке г) дошло до суштинских губитака функцијских зависности и којих?

Одговор:

TS→A, EF→ST