Napredni modeli i baze podataka – ispit 08.09.2016.

U zadacima 1, 2, 3 i 4 se pretpostavlja korištenje PostgreSQL SUBP-a i baze podataka sa slike 1.

Segment baze podataka prikazan na **slici 1** služi za evidenciju podataka o studijskim boravcima studenata na inozemnim visokim učilištima u okviru Erasmus+ programa. Relacija ***visokoUciliste*** sadrži sva sveučilišta i njima podređene visokoobrazovne institucije (fakultete, visoke škole,…) koje sudjeluju u programu. Sveučilišta nemaju nadređeno visoko učilište. Za studijske boravke se bilježi broj mjeseci koje je student boravio na inozemnoj instituciji (**erasmusBoravak.brMjes**).
U relaciji ***ugovorAkGodina*** su pohranjeni podaci o stranim visokim učilištima s kojima visoka učilišta imaju sklopljene ugovore o razmjeni studenata u određenoj akademskoj godini. Iznos mjesečne potpore/stipendije (relacija ***potpora***) koja se studentu dodjeljuje, ovisi o državi u kojoj student boravi za vrijeme razmjene i vremenom se mijenja. Ključevi relacija su podcrtani.

Slika 1.

drzava

Ozn Drzava	***nazivDrzava***	***geom***
HR	Republika Hrvatska	<polygon>
SE	Kraljevina Švedska	<polygon>
PL	Poljska	<polygon>
…	…	

visokoUciliste

sifVU	***nazivVU***	***Ozn Drzava***	***sifNadVU***	***geom***
10	Sveučilište u Zagrebu	HR	NULL	<point>
15	Sveučilište u Splitu	HR	NULL	<point>
53	Fakultet elektrotehnike i računarstva	HR	10	<point>
54	Ekonomski fakultet	HR	10	<point>
112	Kraljevsko tehničko sveučilište	SE	NULL	<point>
…	…	…	…	

erasmusBoravak

JMBAG	***Ak Godina***	**sifVU Odl**	***sifVU Dol***	***brMjes***
…	2015	*53*	*112*	*5*
…	2015	54	117	10
…	…	…	…	…

potpora

ak Godina	***oznDrz Odl***	***oznDrz Dol***	***mjIzn Eur***
2015	HR	SE	400
2016	HR	SE	460
2015	SI	SE	500
…		…	…

ugovorAkGodina

ak Godina	***sifVU Odl***	***sifVUDol***
2015	*53*	[112, 345, …]
2015	*54*	[113, 114, …]
…	…	…

1. **(4 boda)** Za visoka učilišta čiji studenti su sudjelovali u Erasmus+ programu, po akademskim godinama u kojima su boravci realizirani, ispisati ukupan i kumulativan broj studenata te ukupan i kumulativan iznos dodijeljenih potpora/stipendija.
   Npr. kumulativni broj studenata za FER u 2015./2016. akademskoj godini pored broja studenata u 2015./2016. uključuje i broj studenata za godinu 2014./2015. te brojeve studenata u svim godinama prije 2015./2016. Slično vrijedi za kumulativan iznos.

akGodina	nazivVU	brojStud	kumulBrojStud	ukIznosEur	kumulUkIznosEur
…	***…***	***…***	***…***	***…***	***…***
2015	Fakultet elektrotehnike i računarstva	50	200	10000	50000
2016	Fakultet elektrotehnike i računarstva	60	260	12000	62000
…	…	…	…	…	…

2. **(3 boda)** Ispisati nazive svih stranih visokih učilišta (bez ponavljanja) s kojima, u akademskoj godini 2015./2016, bilo koje hrvatsko visoko učilište ima sklopljen ugovor o razmjeni studenata.

3. **(4 boda)** Vlasnik poslovnog sustava koji koristi segment baze podataka sa slike 1 ne želi evidentirati iznose financijskih potpora koje se isplaćuju studentima svake akademske godine nego samo kad se promijene. Smatrati da se iznosi potpora ne mijenjaju unutar akademske godine.
   Potrebno je predložiti izmjene modela sa slike 1 tako da omogući praćenje promjene atributa ***potpora.mjIznosEur*** u kontekstu vremena valjanosti. Modelom osigurati jedinstvenost iznosa potpore za određenu odlaznu i dolaznu državu u konkretnom periodu valjanosti.
   Korištenjem funkcionalnosti PostgreSQL SUBP, napisati SQL naredbe za provođenje predloženih izmjena. Definirati temporalna integritetska ograničenja koja je moguće definirati u PostgreSQL SUBP. Skicirajte sadržaj relacije ***potpora*** nakon predloženih izmjena uz pretpostavku da u njoj postoje samo n-torke prikazane na slici 1. Pretpostavite da iznosi potpora važeći u 2016./2017. vrijede i u 2017./2018., ali ne i nakon. Smatrati da akademska godina počinje 1.10. a završava 30.9. (sljedeće kalendarske godine).
   Napišite jednu SQL naredbu čije bi izvođenje završilo pogreškom zbog narušenja ograničenja temporalnog primarnog ključa. Na primjeru predložene SQL naredbe objasnite na koji način PostgreSQL SUBP (logički) provjerava integritet temporalnog primarnog ključa.

4. Pretpostavimo da je vaš zadatak analizirati i vizualizirati podatke o Erasmus+ programu. Ovdje nećemo doista vizualizirati već analizirati kako organizirati i prirediti podatke odgovarajućeg tipa potrebne za vizualizaciju. Rješenja pišite u formi pseudokoda i SQL naredbi, te komentara, pri čemu **navedite GIS funkcije koje planirate koristiti.**

Potrebno je:

(a) (**3 boda**) Označiti različitim bojama države s obzirom na razmjenu s ostalim državama, svih vremena. Kao mjeru razmjene uzeti broj razmijenjenih studenata (smjer razmjene nije bitan). Na primjer, ako je Hrvatska u cijeloj svojoj povijesti razmijenila s Poljskom 100 studenata, Švedskom 200, te Portugalom 30, suradnja Hrvatske je 330. Primijetite da se, npr., i Poljskoj broji tih istih 100 studenata, odnosno da se razmijenjeni studenti broje i jednoj i drugoj državi.
Opišite kako biste napravili takvu vizualizaciju.
Kod dohvata podataka je potrebno navesti odgovarajuće SQL naredbe.

(b) (**4 boda**) Za svako sveučilište u Erasmus+ programu odrediti njemu najbliže sveučilište iz druge države. Pretpostavite da za svako sveučilište postoji samo jedno najbliže sveučilište.
Za ona sveučilišta:

- koja su u simetričnom odnosu (međusobno su jedna drugom najbliža)
- i čije države graniče

na karti je potrebno označiti poligon koji obuhvaća sva njihova visoka učilišta.
Napišite pseudokod kojim biste to ostvarili, pritom koristite SQL za opis dohvata podataka, pri čemu navedite i GIS funkcije koje koristite.

5. (4 boda) Objasnite zbog čega standardni indeks koji koristi strukturu balansiranog stabla nije pogodan za pretraživanje teksta (full text search) u RSUBP. Objasnite na koji način funkcionira invertirani indeks i zbog čega predstavlja bolje rješenje od standardnog indeksa u kontekstu pretraživanja teksta.

6. (5 bodova) Napisati MapReduce upit koji će na temelju upita koji spaja tablice erasmusBoravak i visokoUciliste vratiti ukupan broj godina i mjeseci ostvarene suradnje između svakog para sveučilišta tako da npr. možemo popuniti sljedeću tablicu:

sveučilite 1	sveučilište 2	godina	mjeseci
Kraljevsko tehničko sveučilište	Sveučilište u Zagrebu	13	9
Kraljevsko tehničko sveučilište	Sveučilište u Splitu	15	0
...	...	...	...

Svaki par ispisati samo jednom.
Pretpostaviti istu M/R arhitekturu (s obzirom na *(combinable) reducer* i sl.) kao MongoDB.

7. (**3 boda**) Što je konzistentno raspršenje? Navedite primjer.

Rješenja:

1. (4 boda)

```
SELECT erasmusBoravak.akGodina, VUOdlazno.nazivVU
     , COUNT(erasmusBoravak.JMBAG)
     , SUM(COUNT(erasmusBoravak.JMBAG)) OVER (PARTITION BY VUOdlazno.nazivVU
                                              ORDER BY erasmusBoravak.akGodina
                                              ROWS BETWEEN UNBOUNDED PRECEDING
                                                       AND CURRENT ROW)
//prethodna 2 retka se mogu izostaviti jer je to i defaultna definicija granica okvira
     , SUM(brMjes*mjIznosEur)
     , SUM(SUM(brMjes*mjIznosEur))   OVER (PARTITION BY VUOdlazno.nazivVU
                                              ORDER BY erasmusBoravak.akGodina
                                              ROWS BETWEEN UNBOUNDED PRECEDING
                                                       AND CURRENT ROW)
//prethodna 2 retka se mogu izostaviti
  FROM erasmusBoravak
     , visokoUciliste VUOdlazno
     , visokoUciliste VUDolazno
     , potpora
 WHERE erasmusBoravak.sifVUOdl = VUOdlazno.sifVU
   AND erasmusBoravak.sifVUDol = VUDolazno.sifVU
   AND VUODlazno.oznDrzava = potpora.oznDrzavaOdl
   AND VUDolazno.oznDrzava = potpora.oznDrzavaDol
   AND erasmusBoravak.akGodina = potpora.akGodina
   GROUP BY erasmusBoravak.akGodina, VUOdlazno.nazivVU
```

2. (3 boda)

```
SELECT DISTINCT stranoVU.nazivVU
FROM ugovorAkGodina,
     UNNEST(sifVuDol) AS sifStranoVU,
     visokoUciliste hrVU,
     visokoUciliste stranoVU
WHERE sifStranoVU =  stranoVU.sifVU
AND ugovorAkGodina.akGodina = 2015
AND ugovorAkGodina.sifVuOdl = hrVU.sifVU
AND hrVU.oznDrzava = 'HR' -- ili spojiti još s državom, ali to nije važno

ili

SELECT DISTINCT stranoVU.nazivVU
FROM ugovorAkGodina,
     visokoUciliste hrVU,
     visokoUciliste stranoVU
WHERE ugovorAkGodina.akGodina = 2015
AND ugovorAkGodina.sifVuOdl = hrVU.sifVU
AND hrVU.oznDrzava = 'HR'
AND sifVUDol @> ARRAY[stranoVU.sifvu]
```

3. (4 boda)

Shemu ***potpora*** treba izmijeniti – atribut akGodina SMALLINT treba zamijeniti atributom akGodinaPeriod tipa DATERANGE (prvo dodati novi atribut, ažurirati mu vrijednost u skladu s vrijednošću akGodina, potom ukloniti akGodina), uništiti postojeći PRIMARY KEY i kreirati temporalni primarni ključ.

```
CREATE EXTENSION bTree_gist; //studenti ne moraju napisati, ali inače treba
```

```
ALTER TABLE potpora DROP CONSTRAINT pkPotpora;
ALTER TABLE potpora ADD akGodinaPeriod DATERANGE;
```
Ažurirati vrijednosti … UPDATE naredbe

```
ALTER TABLE potpora ADD  CONSTRAINT pkPotpora
                    PRIMARY KEY (akGodinaPeriod, oznDrzavaOdl, oznDrzavaDol); //ovo još
uvijek ne osigurava temporalni PK, samo osigurava jednistvenost ove tri atributa
ALTER TABLE potpora ADD CONSTRAINT temporalPKPotpora EXCLUDE USING gist
 (oznDrzavaOdl WITH =, oznDrzavaDol WITH =, akGodinaPeriod WITH &&)
```

Sadržaj izmijenjene relacije:

akGodinaPeriod	***oznDrzavaOdl***	***oznDrzavaDol***	***mjIznosEur***
[1.10.2015, 1.10.2016)	*HR*	*SE*	400.00
[1.10.2016, 1.10.2018)	*HR*	*SE*	460.00
[1.10.2015, 1.10.2016)	*SI*	*SE*	500.00

Zbog u Postgresu podrazumijevane open-closed notacije za DATERANGE tip podatka ispravno je napisati periode [1.10.2015, 1.10.2016), a ne [1.10.2015, 30.9.2016].

Netrivijalna naredba koja bi narušila constraint temporalPKPotpora:
`INSERT INTO potpora VALUES ('[1.12.2015, 1.3.2016)', 'HR', 'SE', 420.00);`
PostgreSQL pri provjeri `constraint temporalPKPotpora` provjerava postoji li u potpora n-torka sa istim `oznDrzavaOdll i oznDrzavaDol (zbog oznDrzavaOdl WITH =, oznDrzavaDol WITH =)` i
periodom koji se preklapa s periodom n-torke koju želimo insertirati (akGodinaPeriod WITH &&). Budući da takav period postoji (n-torka broj 1) sustav neće dozvoliti unos.

4.

(a)

```
// priznaje se i grupiranje po geom, inače trebalo bi napraviti podupit u select
// listi ili sl.
CREATE TEMP TABLE suradnja AS
SELECT drzava.OznDrzava, nazDrzava, geom, COUNT(*) as promet
  FROM visokoUciliste
   JOIN drzava
     ON visokoUciliste.OznDrzava = drzava.OznDrzava
   JOIN erasmusBoravak
     ON (sifVuOdl = sifVu OR sifVuDol = sifVu)
 GROUP BY drzava.OznDrzava, nazDrzava, geom
```

Ovaj upit je ulaz u alat za vizualizaciju kao QGIS, gdje onda treba jednostavno dodijeli boja na temelju mjere „promet“.

(b)

```
CREATE VIEW sveucilista AS
SELECT sv1.sifVu, sv1.oznDrzava,
       sv2.sifVu as sifVu2, sv2.oznDrzava as oznDrzava2,
       st_distance(sv1.geom, sv2.geom) as distance
  FROM visokoUciliste sv1
  JOIN visokoUciliste sv2 ON sv1.oznDrzava <> sv2.oznDrzava -- suvišno
 WHERE sv1.sifNadVu IS NULL
   AND sv2.sifNadVu IS NULL
   AND st_distance(sv1.geom, sv2.geom) =
             (SELECT MIN(st_distance(sv1.geom, sv2.geom))
                FROM visokoUciliste sv3
               WHERE sv1.oznDrzava <> sv3.oznDrzava
                 AND sv3.sifNadVu IS NULL
```

```
            )
CREATE VIEW sim_parovi AS
  SELECT s1.sifVu, s1.sifVu2
    FROM sveucilista s1
    JOIN sveucilista s2 ON  s1.sifVu2 = s2.sifVu
                        AND s2.sifVu2 = s1.sifVu
    JOIN drzava d1 ON s1.oznDrzava = d1.oznDrzava
    JOIN drzava d2 ON s2.oznDrzava = d2.oznDrzava
  WHERE st_touches(d1.geom, d2.geom);

foreach (par in sim_parovi) {
  LET geom = SELECT ST_ConvexHull(ST_Collect(visokoUciliste.geom))
             FROM visokoUciliste
            WHERE sifNadVu in (par.sifVu, par.sifVu2)
  // draw geom
}
```

5. (4 boda)

Pomoću standardnog indeksa koji koristi strukturu B stabla, indeksa, zbog ugrađenih algoritma pretraživanja moguće je obavljati samo upite koji traže zapise s točno jednakim sadržajem atributa nad kojim je indeks izgrađen ili zapise koji imaju na početku atributa traženi uzorak. Npr.:

`SELECT * FROM book WHERE content LIKE 'Once%';` **koristi indeks**

`SELECT * FROM book WHERE content LIKE '%Once %';` **NE koristi indeks**

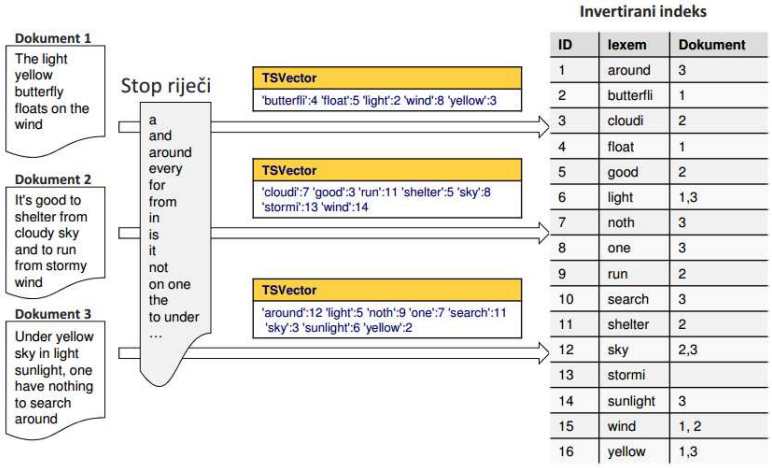
Takvi upiti nisu realni pa ni indeks ne pomaže. Dodatno, ako promotrimo gornji primjer, atribut s `book.content` će vrlo vjerojatno biti različit za svaku n-torku (koliko ima knjiga identičnog sadržaja?). To znači da će taj neupotrebljivi indeks vjerojatno biti UNIQUE i zauzimati puno prostora.

Pomoću ovakvog indeksa nije moguće obaviti pretraživanje koje bi vodilo računa o:

- stop riječima,
- različitim pojavnim oblicima riječi s istim korijenom/osnovom,
- različitim riječima jednakog značenja,

kao niti pretraživanje koje bi primjenjivalo neku od tehnika približne pretrage teksta kao što je npr. pretraga pomoću Q-gram algoritama.

Invertirani indeks s logičkog stanovišta, za svaku riječ koja se pojavljuje u svim vrijednostima atributa nad kojim je definiran, sadrži identifikatore n-torki koji tu riječ sadrže. To omogućuje brzo pronalaženje zapisa koji sadrže kombinaciju nekih riječi tj. određeni uzorak. RSUBP kao što je Postgres koji imaju ugrađenu potporu za normalizaciju teksta, omogućavaju izgradnju indeksa nad normaliziranom verzijom sadržaja atributa pa time invertirani indeks postaje upotrebljiv za pretragu koja vodi računa o stop riječima, različitim pojavnim oblicima riječi s istim korijenom/osnovom, različitim riječima jednakog značenja i td.



Invertirani indeks

ID	lexem	Dokument
1	around	3
2	butterfli	1
3	cloudi	2
4	float	1
5	good	2
6	light	1,3
7	noth	3
8	one	3
9	run	2
10	search	3
11	shelter	2
12	sky	2,3
13	stormi	
14	sunlight	3
15	wind	1, 2
16	yellow	1,3

6.

```
map (k, v) {
   // poredamo ih, abecedno ili po šifri, svejedno, bitno je da je konzistentno
   // ideja je da se i (33, 35) i (35, 33) emitiraju kao jedan ključ, npr. (33, 35)
```

```
   Sv1 = MIN(k.nazivSveuc1, k.nazivSveuc2)
   Sv2 = MAX(k.nazivSveuc1, k.nazivSveuc2)
       emit({Sv1, Sv2}, k.brMjes);
}
reduce (k, vlist) {
// ovdje se ne smije pretvarati u godine, jer je CR
// zapravo, dalo bi se, ali nije elegantno
   var sumMj = 0;
   foreach (v in vlist) {
      sumMj += v;
   }
   return sumMj;
}
finalize (k, v) {
   return { k.Sv1, k.Sv2, v / 12, v % 12 };
}
```

6. Vidi predavanja.