Tim Hajdinjak, 63230099

Seminarska naloga PB za leto 2024

1. Naloga

Iz tabele x_world, opisane z relacijsko shemo:

x_world(id, x, y, tid, vid, village, pid, player,
aid, alliance, population)

naredite (CREATE TABLE) in napolnite tabele z naslednjimi relacijskimi shemami in pomeni. Pozorni bodite na primarne in tuje ključe ter pravilni vrsti red ustvarjanja tabel.

Rešitev:

```
CREATE TABLE pleme(
    tid int,
    tribe varchar(100),
    PRIMARY KEY(tid)
);


CREATE TABLE aliansa(
    aid int,
    alliance varchar(100),
    PRIMARY KEY(aid)
);


CREATE TABLE igralec(
    pid int,
    player varchar(100),
    tid int,
    aid int,
    FOREIGN KEY(tid) REFERENCES pleme(tid) ON UPDATE CASCADE ON
DELETE CASCADE,
    FOREIGN KEY(aid) REFERENCES aliansa(aid) ON UPDATE CASCADE ON
DELETE CASCADE,
    PRIMARY KEY(pid)
```

```
);


CREATE TABLE naselje(
    id int,
    x int,
    y int,
    vid int,
    village varchar(100),
    population int,
    pid int,
    FOREIGN KEY(pid) REFERENCES igralec(pid) ON UPDATE CASCADE ON
DELETE CASCADE
);


INSERT INTO pleme VALUES
(1,'Rimljani'),
(2,'Tevtoni'),
(3,'Galci'),
(4,'Narava'),
(5,'Natarji');


INSERT INTO aliansa (SELECT DISTINCT aid, alliance FROM x_world
where aid != 0);
INSERT INTO igralec (SELECT DISTINCT pid, player, tid, aid FROM
x_world WHERE tid NOT IN(6,7) AND aid != 0); /*V navodilih piše samo
od 1-5, 6 pa 7 damo ven*/
INSERT INTO naselje (SELECT DISTINCT id, x, y, vid, village,
population, pid FROM x_world WHERE tid NOT IN(6,7) AND aid != 0);
/*enak razlog kot zgoraj (drugače, če navodilo to ni zahtevalo,
preprosto damo WHERE pogoj pri obeh poizvedbah ven, ter dodamo pleme
6 pa 7 v tabelo pleme)*/
```

Komentar: kot sem pri zadnjih dveh INSERT stavkih zapisal, sem omejil definicijsko območje plemen na tid = [1,5], saj sem tako razumel preko navodil. V kolikor sem napačno razumel, se v resnici spremeni samo zaloga vrednosti (rezultati) poizvedb, sama logika pa vseeno ostane enaka (to pa v bistvu zares šteje 😊).

2. Naloga

- Naredite pogled x_view, ki bo iz novih tabel naredil pogled ekvivalenten originalni tabeli x_world (CREATE VIEW).

- S pomočjo SQL poizvedb smiselno preverite, ali sta vsebini x_view in x_world identični.

- *S pomočjo ustreznih DDL ukazov ustvarite tabelo top5(alliance, SteviloNaselij), ki hrani alianse z največ naselji. Dodajte bazne prožilce tako, da se bo lista posodabljala vsakič, ko nekdo zgradi novo naselje.

Rešitve:

```
/*a*/

CREATE VIEW x_view AS(

    SELECT n.id, n.x, n.y, i.tid, n.vid, n.village, i.pid, i.player, a.aid, a.alliance, n.population

    FROM igralec i inner join pleme p on(p.tid=i.tid) inner join aliansa a on(a.aid=i.aid) inner join naselje n on(n.pid=i.pid)

);

/*b*/

SELECT IF((SELECT COUNT(*) FROM x_world) = (SELECT COUNT(*) FROM x_view),"Sta","Nista") AS "Ali sta tabeli enaki?";

/*vemo, da sta strukturno enaki in da nekatera vsebina je enaka (za plemena od 1-5), zato samo preverimo, če se razlikujeta v številu vrstic*/
```

Rezultat:

```
+----------------------+

| Ali sta tabeli enaki? |

+----------------------+

| Nista                |

+----------------------+
```

```
/*c*/
CREATE TABLE top5 AS(
    SELECT a.alliance, COUNT(*) AS SteviloNaselij
    FROM aliansa a inner join igralec i on(i.aid=a.aid) inner join
naselje n on(n.pid=i.pid)
    GROUP BY a.alliance
    ORDER BY SteviloNaselij DESC
    LIMIT 5
);
```

Rezultat: (SELECT * FROM top5;)

```
+----------+----------------+
| alliance | SteviloNaselij |
+----------+----------------+
| DARK?    |            303 |
| RS-SN    |            289 |
| RS-H1N1? |            284 |
| DT RS #2 |            259 |
| DLZ SOS  |            238 |
+----------+----------------+
/*bazni sprožilci za top5*/
DELIMITER //
CREATE TRIGGER posodobiTop5_Insert
AFTER INSERT ON naselje
FOR EACH ROW
BEGIN
    DELETE FROM top5;
    INSERT INTO top5 (
        SELECT a.alliance, COUNT(*) AS SteviloNaselij
        FROM aliansa a inner join igralec i on(i.aid=a.aid) inner
join naselje n on(n.pid=i.pid)
        GROUP BY a.alliance
```

```
        ORDER BY SteviloNaselij DESC
        LIMIT 5
    );
END //
DELIMITER ;

DELIMITER //
CREATE TRIGGER posodobiTop5_Update
AFTER UPDATE ON naselje
FOR EACH ROW
BEGIN
    DELETE FROM top5;
    INSERT INTO top5 (
        SELECT a.alliance, COUNT(*) AS SteviloNaselij
        FROM aliansa a inner join igralec i on(i.aid=a.aid) inner
join naselje n on(n.pid=i.pid)
        GROUP BY a.alliance
        ORDER BY SteviloNaselij DESC
        LIMIT 5
    );
END //
DELIMITER ;

DELIMITER //
CREATE TRIGGER posodobiTop5_Delete
AFTER DELETE ON naselje
FOR EACH ROW
BEGIN
    DELETE FROM top5;
    INSERT INTO top5 (
```

```
        SELECT a.alliance, COUNT(*) AS SteviloNaselij
        FROM aliansa a inner join igralec i on(i.aid=a.aid) inner
join naselje n on(n.pid=i.pid)
        GROUP BY a.alliance
        ORDER BY SteviloNaselij DESC
        LIMIT 5
    );
END //
DELIMITER ;
```

Komentar: Predvidevam da ste hoteli samo za INSERT, vendar za UPDATE pa DELETE ne škodi če dodam, ker je enaka logika.

3. Naloga

a) Kateri igralec ima največje naselje?

b) Koliko igralcev ima nadpovprečno veliko naselje?

c) Izpišite podatke o vseh naseljih igralcev brez alianse, urejeno padajoče po x in nato y koordinati.

d) Katero pleme je najštevilčnejše (glede na populacijo)?

e) Izpišite število nadpovprečno močnih alians (povprečje populacije računajte glede na alianse, ne na vse igralce).

f) *Igralec bananamen želi preimenovati vsa svoja naselja na naslednji način. Uredil jih bo po populaciji, najmočnejše bo »Banana-00«, naslednje »Banana-01« in tako dalje. Nalogo lahko rešite v več korakih (zaporedju poizvedb).

g) Napišite shranjen podprogram, ki za poljubne koordinate (parametra x in y) vrne populacijo na največ podani razdalji (parameter razdalja). Npr. razdalja 10 pomeni vse koordinate od vključno (x-10, y-10) do (x+10, y+10). Za preverjanje robnih pogojev (koordinate izven [-400,400] po potrebi uporabite IF stavek.

h) Izpišite imena igralcev, ki imajo vsa svoja naselja na območju x, ki je med 100 in 200 in y, ki je med 0 in 100.

i) Napišite poizvedbo, ki bo poiskala vsa naselja s populacijo točno 1000. Napišite še stavek, ki bi to poizvedbo pohitril, če bi bila tabela naselje dovolj velika.

j) Poiščite igralce, ki imajo umirajoče naselje. Za umirajoče naselje vzemite tista naselja, ki imajo manj kot 3% povprečne populacije igralca (povprečna populacija igralca je populacija igralca ulomljeno s številom njegovih naselij).

Rešitve:

```
/*a*/
SELECT i.player AS "Najvecje naselje ima:"
FROM igralec i INNER JOIN naselje n ON(n.pid=i.pid)
WHERE n.vid IN(
    SELECT nas.vid
    FROM (
        SELECT vid, MAX(population)
        FROM naselje
    ) AS nas
);
```

Rezultat:

```
+----------------------+
| Najvecje naselje ima: |
+----------------------+
| tecec                |
+----------------------+
```

```
/*b*/
SELECT COUNT(*) AS "Stevilo igralcev z nadpovprecnim naseljem:"
FROM igralec
WHERE pid IN(
    SELECT i.pid
    FROM igralec i INNER JOIN naselje n ON(i.pid=n.pid)
    GROUP BY i.pid
    HAVING SUM(n.population) >= (SELECT AVG(population) FROM
naselje)
);
```

Rezultat:

```
+-----------------------------------------+
| Stevilo igralcev z nadpovprecnim naseljem: |
+-----------------------------------------+
|                                     979 |
+-----------------------------------------+
```

```
/*c*/
SELECT n.*
FROM igralec i INNER JOIN naselje n ON(n.pid=i.pid)
WHERE i.aid = 0
ORDER BY n.x DESC, n.y DESC;
```

Rezultat:

Vrne nič, ker alianse z aid = 0, ni v tabeli alians (takšno je bilo navodilo).

```
/*d*/
SELECT p.tribe AS "Najstevilcnejse pleme je:"
FROM pleme p INNER JOIN igralec i ON(i.tid=p.tid) INNER JOIN naselje n ON(n.pid=i.pid)
GROUP BY p.tribe
HAVING SUM(n.population) = (
    SELECT plem.skupno
    FROM (
        SELECT p.tid, SUM(n.population) AS skupno
        FROM pleme p INNER JOIN igralec i ON(i.tid=p.tid) INNER JOIN naselje n ON(n.pid=i.pid)
        GROUP BY p.tid
        ORDER BY SUM(n.population) DESC
        LIMIT 1
    ) AS plem
```

```
);
/*v primeru, če sta dva plemena z isto populacijo, vrne oba (detajli
:) )*/
```

Rezultat:

```
+---------------------------+
| Najstevilcnejse pleme je: |
+---------------------------+
| Rimljani                  |
+---------------------------+
```

```
/*e*/
SELECT COUNT(*) AS StAlians
FROM aliansa
WHERE aid IN(
    SELECT a.aid
    FROM aliansa a INNER JOIN igralec i ON(a.aid=i.aid) INNER JOIN
naselje n ON(n.pid=i.pid)
    GROUP BY a.aid
    HAVING SUM(n.population) >= (
        SELECT AVG(tab.skupno)
        FROM (
            SELECT a.aid, SUM(n.population) as skupno
            FROM aliansa a INNER JOIN igralec i ON(a.aid=i.aid)
INNER JOIN naselje n ON(n.pid=i.pid)
            GROUP BY a.aid
        ) AS tab
    )
);
```

Rezultat:

```
+----------+
| StAlians |
+----------+
|       39 |
+----------+
```

```
/*g*/
DELIMITER //
CREATE PROCEDURE populacijaNaRazdalji(IN x INTEGER, IN y INTEGER, IN razdalja INTEGER, OUT populacija INTEGER)
BEGIN
    /*A ni malce brezveze preverjat ce je izven obmocja? Ker ce ne obstaja bo vrnil 0?*/
    IF (x NOT BETWEEN -400 AND 400) OR (y NOT BETWEEN -400 AND 400) THEN
        SIGNAL SQLSTATE '42000' SET MESSAGE_TEXT = 'Error: x in y morata imeti zalogo vrednosti: [-400, 400]';
    ELSE
        SELECT SUM(n.population) INTO populacija
        FROM naselje n
        WHERE n.x BETWEEN x-razdalja AND x+razdalja AND n.y BETWEEN y-razdalja AND y+razdalja;
    END IF;
END //
DELIMITER ;
CALL populacijaNaRazdalji(5,5,10,@pop);
SELECT @pop;
```

Rezultat:

```
+------+
| @pop |
+------+
| 5680 |
+------+
```

```
/*h*/
SELECT DISTINCT i.player
FROM igralec i
WHERE i.pid IN(
    SELECT i.pid
    FROM igralec i INNER JOIN naselje n ON(n.pid=i.pid)
    WHERE n.x BETWEEN 100 AND 200 AND n.y BETWEEN 0 AND 100
) AND i.pid NOT IN(
    SELECT i.pid
    FROM igralec i INNER JOIN naselje n ON(n.pid=i.pid)
    WHERE n.x NOT BETWEEN 100 AND 200 AND n.y NOT BETWEEN 0 AND 100
);
```

Rezultat:

```
+-----------------+
| player          |
+-----------------+
| ???             |
| ASOPDIAS        |
| TotalnoPoludeli |
| mataba          |
| mrazdeda        |
| panzer          |
| ...(se nadaljuje)|
```

```
+------------------+
```

```
/*i*/
/*ni cisto jasno kaj naj poizvedba tocno vrne*/
SELECT n.*
FROM naselje n
WHERE n.population = 1000;

CREATE INDEX poPopulaciji ON naselje(population);
```

Rezultat:

Vrne nič, takšnih naselij nad takimi podatki ni.

```
/*j*/
SELECT i.player
FROM igralec i INNER JOIN naselje n ON(n.pid=i.pid)
WHERE n.population <= (
    SELECT povp * 0.03
    FROM (
        SELECT i2.pid AS igralc, AVG(n2.population) AS povp
        FROM igralec i2 INNER JOIN naselje n2 ON(n2.pid=i2.pid)
        GROUP BY igralc
    ) AS povpIgralca
    WHERE igralc = i.pid
);
```

Rezultat:

```
+-----------------+
| player          |
+-----------------+
| japy            |
| futrač          |
| Robin Hood      |
| bez             |
| Serafina Pekala |
| acko            |
|. . . (se nadaljuje) |
+-----------------|
```

4. Naloga

V programskem jeziku Python napišite program, ki se priključi na podatkovno bazo in za celotno igralno polje izračuna gostoto populacije in gostoto populacije določene alianse.

Gostoto računajte na območjih velikosti 10x10 polj po formulah:

$$Gostota\ populacije = Skupna\ populacija\ na\ območju/100$$

$$Gostota\ alianse = Skupna\ populacija\ alianse\ na\ območju/100$$

Rezultate izračunane gostote (za vsako izmed 80x80=6400 območij) shranite v primerne tabeli gostotaPopulacije in gostotaAlianse. Za alianso izberite najmočnejšo alianso glede na populacijo.

Rešitev:

Rešitev naloge (Python kodo) imate v priloženi datoteki.

Nalogo sem rešil tako, da sem se najprej povezal na fakultetni strežnik, ustvaril kurzorja ter ustvaril tabeli gostotaPopulacije ter gostotaAlianse. Obe tabeli sprejemata atribute: gostota, ter x in y koordinato začetka območja. Preden sem za vsako območje izračunal gostoto, sem poiskal najmočnejšo alianso glede na populacijo. S pomočjo dveh zank sem nato poiskal gostoto celotne populacije ter določene alianse na določenem območju, ter to gostoto

zapisal v pripadajoči tabeli. Opazili boste, da je v obeh tabelah več kot 6400 zapisov (ter tudi območij). To je zaradi tega, ker je med -400 in -391 10 potencialnih območij, medtem ko jih je med -400 in -390 11. Ko iteriraramo do končnih mej koordinat (tj. 400), dobimo predzadnje območje 390 do 399. Ker se **lahko** zgodi, da kakšen igralec zgradi tam svoje naselje, sem v obe tabeli dodal še robno območje (400 do 409, ker od 400+ ni ničesar, SUM vrača nič, s tem ni ničesar narobe).

5. Naloga

Iz programa Microsoft Excel ali LibreOffice Calc (najprej uporabite LibreOffice Base za ODBC povezavo) se priključite na podatkovno bazo in v obliki grafov izrišite rezultate izračunane gostote poselitev iz četrte naloge.

Rešitev:

Preko Podatki->Dobi podatke->Iz drugih virov->Iz vmesnika ODBC sem prejel podatke iz tabel gostotaPopulacija ter gostotaAlianse. Preden sem izrisal graf obeh tabel, sem ju moral pretvoriti v »pivot table«, to sem pa dosegel s tem, da sem pod Vstavljanje->Vrtilna tabela, definiral ustrezne koordinate in vrednosti tabele. S tem smo nato več kot pol dela že opravili, preostane nam samo še kreacija grafov, ki ga dobimo preko Vstavljanje->Priporočeni grafikoni->Stolpčni->3D stolpčni. Končni rezultat (katerega tudi prilagam v priloženi datoteki:)