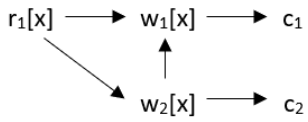
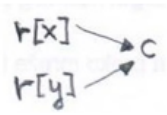
Laboratorij profila II, ak. god. 2020/21.
Odjeljak predmeta Sustavi baza podataka

8. VJEŽBA

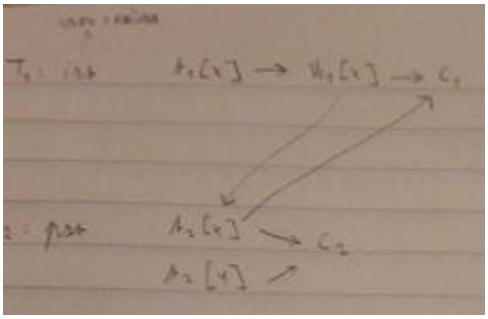
Napomena: Za sve zadatke, u svim vježbama, u kojima treba priložiti sliku grafa ili nečeg sličnog, vrijedi sljedeće: slika se može nacrtati bilo kojim alatom za crtanje ili se može nacrtati na bijelom papiru bez crta ili kvadratića, pa skenirati ili fotografirati. Međutim, nekvalitetne slike (previše tamne, previše blijede, neuredne, nečitkog rukopisa i slično) neće se ocjenjivati, odnosno smatrat će se da nisu dostavljene. U nastavku su prikazani primjeri prihvatljivih i neprihvatljivih slika grafova.

prihvatljivo:





neprihvatljivo:



1. zadatak

Procjena potrebnog vremena: 15 minuta

U rješenjima je potrebno dostaviti:

- Rješenja zadataka 1.1. - 1.3.

Preuzeti datoteku i izvršiti skripta `labprof8.sql`.

Otvoriti četiri SQL Editor kartice (*tab*), a u svakoj od kartica korisničku sjednicu (*session*) za korisničko ime "*sa*" i bazu podataka labprof8, te u svakoj od tih korisničkih sjednica obaviti naredbu `SET CONTEXT_INFO 7;`

> Radi pojednostavljenja teksta, korisničke sjednice će se u nastavku teksta referencirati "mnemoničkim" nazivima: sessionA, sessionB, sessionC, itd. Npr. korisnička sjednica otvorena u prvom od četiri SQL editor prozora naziva se sessionA. Transakcije koje će se izvršavati u sjednici sessionA treba označavati T_1, transakcije koje će se izvršavati u sjednici sessionB treba označavati T_2, itd.

Otvoriti još jednu (petu) korisničku sjednicu (u daljnjem tekstu, nazivat će se *sessionAdmin*).

SQL Server je svakoj korisničkoj sjednici automatski dodijelio identifikator sjednice (*session id*).

1.1. U svakoj sjednici izvršiti naredbu kojom se može dobiti informacija o identifikatoru (*session id*) tekuće korisničke sjednice (sjednice u kojoj se izvršava ta naredba)? Pribilježiti identifikatore sjednica u otvorenim SQL Editor karticama. Zabilježiti npr.: sessionA=54, sessionB=55, itd.

1.2. U SQL Editor kartici sjednice *sessionAdmin* napisati i testirati naredbu kojom će se dobiti popis svih sjednica (identifikator sjednice, vrijeme otvaranja sjednice i naziv baze podataka koju sjednica trenutačno koristi, ako ju koristi). Pomoć: postoji relacija sys.dm_exec_sessions

1.3. Jednako kao u zadatku 1.2, ali ovog puta u popisu se trebaju nalaziti samo sjednice sessionA - sessionD. Pomoć: u svakoj od sjednica sessionA - sessionD izvršena je naredba `SET CONTEXT_INFO`.

Tijekom nadziranih provjera/vježbi očekuju se sljedeća znanja i vještine:

- Objasniti vlastita rješenja zadataka 1.1 - 1.3.

Pomoćni upiti za zadatke u ovim laboratorijskim vježbama

U ovim laboratorijskim vježbama, za provjeru koji ključevi su postavljeni na koje n-torke relacije **`test`**, može se koristiti sljedeći upit:

```
SET TRANSACTION ISOLATION LEVEL READ UNCOMMITTED;
SELECT request_session_id AS sid
     , CASE WHEN request_mode = 'U' THEN 'S' ELSE request_mode END AS lock_type
     , test.sifra
  FROM sys.dm_tran_locks
  LEFT OUTER JOIN test
    ON sys.fn_PhysLocFormatter(%%physloc%%) = '(' + TRIM(resource_description) + ')'
  WHERE resource_type = 'RID'
    AND request_status = 'GRANT'
    AND request_session_id IN (SELECT session_id FROM sys.dm_exec_sessions where CAST(context_info AS INT) = 7);
```

- sessionAdmin koristi razinu izolacije `READ UNCOMMITTED` jer ona sama ne treba postavljati ključeve. Razine izolacije (`READ UNCOMMITTED`, `REPEATABLE READ`, itd.) u ovoj se vježbi mogu koristiti bez razumijevanja čemu služe, a detaljnije će se razmatrati u sljedećoj vježbi laboratorija profila.
- u rezultatu se prikazuju samo uspješno postavljeni (`= 'GRANT'`) ključevi na n-torkama (`= 'RID'`)
- prikazuju se samo ključevi koji su postavljeni u sjednicama sessionA ... sessionD (`context_info = 7`)
- U ovoj vježbi se razlika između S-ključeva i U-ključeva zanemaruje. U-ključevi (koje MS SQL Server ponekad nepotrebno postavlja umjesto S-ključeva), radi pojednostavljenja se u rezultatu upita prikazuju kao S-ključevi (tome služi `CASE ...`).
- spajanje s relacijom **`test`** se obavlja radi jasnijeg prikaza identifikatora ključeva n-torki koje su zaključane

Za provjeru koja transakcija (sjednica) eventualno čeka zbog ključeva koje je postavila neka druga transakcija (sjednica), može se koristiti sljedeći upit:

```
SELECT session_id AS sid_blocked
     , blocking_session_id AS sid_blocking
FROM sys.dm_os_waiting_tasks
 WHERE blocking_session_id IN (SELECT session_id
                               FROM sys.dm_exec_sessions
                               WHERE CAST(context_info AS INT) = 7);
```

2. zadatak

Procjena potrebnog vremena: 20 minuta

U rješenjima je potrebno dostaviti:

- Rješenja zadataka 2.1. - 2.4.

Napomena: na početku svake transakcije u svim zadacima u ovoj laboratorijskoj vježbi (pri tome, u ovoj vježbi nije obavezno razumijevanja svrha te naredbe), izvršiti naredbu `SET TRANSACTION ISOLATION LEVEL REPEATABLE READ`.

U korisničkoj sjednici sessionA i korisničkoj sjednici sessionB naizmjence obavljati sljedeće naredbe (jednu naredbu u sessionA, jednu naredbu u sessionB, itd., i usput odgovoriti na sljedeća pitanja:

2.1. Čemu služi naredba `SET LOCK_TIMEOUT -1`?

2.2. Koje ključeve je koja transakcija na koje n-torke postavila tijekom izvršavanja svoje naredbe SELECT? Koja transakcija (sjednica) čeka zbog ključeva koje transakcije (sjednice)?

2.3. Koje ključeve je koja transakcija na koje n-torke postavila ili promovirala tijekom izvršavanja naredbi UPDATE? Koja transakcija (sjednica) čeka zbog ključeva koje transakcije (sjednice)? Je li nakon pokušaja izvršavanja tih naredbi nastupio potpuni zastoj?

2.4. X-ključ se ne smije postaviti na objekt koji je već zaključan S-ključem (npr. uočiti da je na n-torku sa šifrom 8 postavljen S-ključ). Zašto je onda T_2 uspjela uspjela izvršiti UPDATE nad tom n-torkom?

T_1 (sessionA)	**T_2 (sessionB)**
`SET LOCK_TIMEOUT -1;`	`SET LOCK_TIMEOUT -1;`
`BEGIN TRANSACTION;`	`BEGIN TRANSACTION;`
`SET TRANSACTION ISOLATION LEVEL REPEATABLE READ;`	`SET TRANSACTION ISOLATION LEVEL REPEATABLE READ;`
`SELECT mjera FROM test WHERE sifra = 5;`	
	`SELECT mjera FROM test WHERE sifra = 8;`
	`UPDATE test SET mjera = 82.0 WHERE sifra = 8;`
`UPDATE test SET mjera = 81.0 WHERE sifra = 8;`	
`ROLLBACK TRANSACTION;`	`ROLLBACK TRANSACTION;`

Tijekom nadziranih provjera/vježbi očekuju se sljedeća znanja i vještine:

- Objasniti vlastita rješenja zadataka 2.1 - 2.4. Obaviti sličan eksperiment i objasniti njegove rezultate.

3. zadatak

Procjena potrebnog vremena: 40 minuta

U rješenjima je potrebno dostaviti:

- SQL naredbu za kreiranje procedure **proc3**
- sve SQL naredbe potrebne za testiranje procedure, s kratkim popratnim objašnjenjem rezultata testova

3.1. Napisati i testirati T-SQL pohranjenu proceduru **proc3** koja n-torkama sa zadanom šifrom (*sifra* je ulazni argument procedure) u sve tri relacije (test, test2 i test3) uvećava vrijednost atributa mjera za 10% (dakle, potrebno je obaviti ukupno tri SQL naredbe UPDATE). Procedura upravlja transakcijom: započinje transakciju, ako se tijekom izvršavanja dogodi pogreška poništava transakciju i dojavljuje pogrešku, inače, potvrđuje transakciju. Na zaključavanje u relaciji test čekati dok se ne dobije ključ - koliko god dugo bude potrebno; na zaključavanje u relaciji test2 čekati dok se ne dobije ključ, ali ne više od 5 sekundi; na zaključavanje u relaciji test3 ne smije se čekati - ako se ključ ne dobije odmah po zahtjevu, smatra se da se dogodila pogreška. Procedura na pogreške treba reagirati na sljedeći način:

- ako se dogodi pogreška zbog predugog čekanja na postavljanje ključa ili neuspješnog postavljanja ključa, dojaviti pogrešku error_number=50501, message='Privremeno zaključano, pokušajte kasnije.', state=1
- ako se tijekom izvršavanja procedure dogodi bilo koja druga pogreška, procedura treba dojaviti originalnu pogrešku koju je producirao sustav za upravljanje bazama podataka.

Tijekom nadziranih provjera/vježbi očekuju se sljedeća znanja i vještine:

- Objasniti svoje rješenje, naredbe za testiranje i rezultate testiranja. Napisati i testirati T-SQL pohranjenu proceduru ili Java program koji obavlja sličnu zadaću, te objasniti dobivene rezultate testiranja.

4. zadatak

Procjena potrebnog vremena: 30 minuta

U rješenjima je potrebno dostaviti:

- Tablicu sa SQL naredbama za zadatak 4.1.
- Rješenja zadataka 4.2. - 4.4.

4.1. Načiniti tablicu, sličnu onoj u 2. zadatku, koja će opisati redoslijed izvršavanja niza operacija za transakcije T_1, T_2, T_3 i T_4, koje se izvršavaju u sessionA do sessionD (dakle, tablica će imati 4 stupca). Pri tome:

- slobodno odabrati nad kojim n-torkama u kojim relacijama (test, test2, test3) će se izvršavati operacije transakcija
- za T_1 i T_4 slobodno odabrati izvršavanje neograničenog broja istih ili različitih SELECT naredbi koje dohvaćaju mjeru za po jednu n-torku sa zadanom šifrom (npr. SELECT mjera FROM test2 WHERE sifra = 5, SELECT mjera FROM test3 WHERE sifra = 2, itd.)
- za T_2 i T_3 slobodno odabrati izvršavanje neograničenog broja istih ili različitih UPDATE naredbi koje mijenjaju mjeru za po jednu n-torku sa zadanom šifrom (npr. UPDATE test3 SET mjera = 7.1 WHERE sifra = 7, UPDATE test SET mjera = 3.3 WHERE sifra = 3, itd.)
- prije izvršavanja posljednje SQL naredbe u nizu, tri transakcije već moraju biti u stanju čekanja. Tijekom izvršavanja posljednje operacije u nizu operacija mora se dogoditi potpuni zastoj u kojem sudjeluju transakcije T_1, T_2 i T_3, ali ne i transakcija T_4 (dakle, T_4, iako u stanju čekanja, ne sudjeluje u potpunom zastoju).

Riješiti sljedeće zadatke:

4.2. Nacrtati WFG u trenutku kada je nastao potpuni zastoj (prije nego je razriješen).

4.3. Što je SUBP poduzeo u cilju razrješenja potpunog zastoja?

4.4. Nacrtati WFG u trenutku kada je sustav razriješio potpuni zastoj.

Tijekom nadziranih provjera/vježbi očekuju se sljedeća znanja i vještine:

- Objasniti vlastita rješenja. Riješiti sličan zadatak, testirati i objasniti rezultate.

5. zadatak

Procjena potrebnog vremena: 30 minuta

U rješenjima je potrebno dostaviti:

- Rješenje zadatka 5.1.

U sustavu se koristi sljedećih 5 vrsta transakcija koje djeluju na n-torke u relaciji test i opisane su pseudokodom:

a) **`postaviMjeru(x, m)`** -- mjeru n-torke x postavi na zadanu vrijednost m

```
write(x, m);
```

b) **`prikaziMjeru(x)`** -- prikaži mjeru n-torke x

```
read(x, p1);
display(p1);
```

c) **`korekcijaMjere(x, m)`** -- mjeru n-torke x promijeni za zadani iznos m

```
read(x, p1);
p1 ← p1 + m;
write(x, p1);
```

d) **`zamijeniMjere(x, y)`** -- zamijeni mjere n-torke x i n-torke y

```
read(x, p1);
read(y, p2);
write(x, p2);
write(y, p1);
```

e) **`prikaziSumu(x, y)`** -- zbroji i prikaži sumu mjera u n-torkama x i y

```
read(x, p1);
read(y, p2);
p3 ← p1 + p2;
display(p3);
```

Riješiti sljedeći zadatak:

5.1. Nacrtati grafove transakcija opisanih pod a) do e). U svaki od grafova transakcija ucrtati najmanji mogući broj lúkova: isključivo one lúkove koji proizlaze iz semantike transakcije ili su nužni da bi se zadovoljila pravila konstrukcije grafa transakcije.

Tijekom nadziranih provjera/vježbi očekuju se sljedeća znanja i vještine:

- Objasniti vlastita rješenja.

6. zadatak

Procjena potrebnog vremena: 20 minuta (za upoznavanje sa zadatkom i priloženim rješenjem)

U rješenjima je potrebno dostaviti:

- ništa

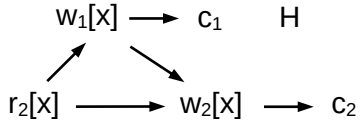
Važno: SQL Server u nekim situacijama ključeve postavlja potpuno izvan pravila 2PL protokola. Eklatantno kršenje 2PL pravila dešava se npr. kada T_1 SELECT naredbom postavi S-ključ na element x. T_2 koja UPDATE naredbom pokušava postaviti X ključ u tome (ispravno) ne uspijeva, ali umjesto da odmah uđe u stanje čekanja, T_2 na x bespotrebno postavlja S-ključ (!) i tek tada uđe u stanje čekanja. Ako T_1 tada pokuša obaviti UPDATE elementa x, dogodit će se potpuni zastoj. To znači da SQL Server u slučaju kada povijest manifestira problem izgubljene izmjene nepotrebno izaziva potpuni zastoj.

Kako bi se u sljedećim zadacima, pri izvođenju eksperimenata, dobili rezultati koji su uglavnom konzistentni s pravilima 2PL protokola, ovo neispravno ponašanje sustava pokušat će se zaobići na sljedeći način: ako transakcija T_1 neki element čita (SELECT), a kasnije će u taj isti element pisati (UPDATE), tada u SELECT naredbu transakcije T_1 obavezno treba dodati opciju WITH (UPDLOCK). Vidjeti primjer u rješenju zadatka 6.1.

Ovo vrijedi samo u vježbama na laboratoriju profila. Na ispitima (npr. završni ispit) se ovo neispravno ponašanje sustava SQL Server ne treba i ne smije uzimati u obzir.

6.1. Nacrtati graf povijesti H koja sadrži transakciju T_1 `postaviMjeru(test.sifra=1, 100.0)` i transakciju T_2 `korekcijaMjere(test.sifra=1, 105.0)`. Povijest mora biti takva da manifestira problem izgubljene izmjene. Zatim načiniti tablicu sličnu tablici u 2. zadatku koja će opisivati redoslijed SQL naredbi za T_1 i T_2, koji je konzistentan s poviješću H. Izvršavanjem SQL naredbi utvrditi stvarni (producirani) redoslijed izvršavanja. Nacrtati graf povijesti H_P koju je sustav producirao.

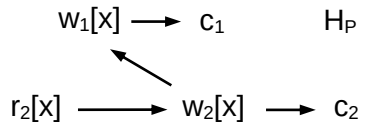
Rješenje:



T_1 (sessionA)	T_2 (sessionB)
`SET LOCK_TIMEOUT -1;`	`SET LOCK_TIMEOUT -1;`
`BEGIN TRANSACTION;`	`BEGIN TRANSACTION;`
`SET TRANSACTION ISOLATION LEVEL REPEATABLE READ;`	`SET TRANSACTION ISOLATION LEVEL REPEATABLE READ;`
	`SELECT mjera FROM test WITH(UPDLOCK) WHERE sifra = 1;` -- rezultat je 10.0, pribroji m=105.0, mjeru postaviti na 115.0 -- isti element kojeg je ovdje pročitala, T_2 će kasnije mijenjati, stoga je u SELECT dodan WITH(UPDLOCK)
`UPDATE test SET mjera = 100.0 WHERE sifra = 1;`	
	`UPDATE test SET mjera = 115.0 WHERE sifra = 1;`
`COMMIT TRANSACTION;`	`COMMIT TRANSACTION;`

Stvarni (producirani) redoslijed izvršavanja: $r_2[x]\ w_2[x]\ c_2\ w_1[x]\ c_1$

Graf povijesti koja je konzistentna s produciranim redoslijedom izvršavanja:



Tijekom nadziranih provjera/vježbi očekuju se sljedeća znanja i vještine:

- Objasniti ovdje priloženo rješenje.

7. zadatak

Procjena potrebnog vremena: 40 minuta

U rješenjima je potrebno dostaviti:

- Rješenja zadataka 7.1. - 7.2.

7.1. a) Nacrtati graf povijesti H koja sadrži transakciju T_1 `prikaziMjeru(test.sifra=4)` i transakciju T_2 `postaviMjeru(test.sifra=4, 410.0)`. Povijest mora biti takva da manifestira problem prljavog čitanja. Objasniti zašto H nije ST (nije striktna).

b) Načiniti tablicu sličnu onoj u 2. zadatku koja će opisivati redoslijed SQL naredbi za T_1 i T_2, koji je konzistentan s poviješću H.

c) Izvršavanjem SQL naredbi utvrditi stvarni (producirani) redoslijed izvršavanja, nacrtati graf povijesti H_P koju je sustav producirao, dokazati da H_P jest CSR i objasniti zašto H_P jest ST (jest striktna).

7.2. a) Nacrtati graf povijesti H koja sadrži transakciju T_1 `zamijeniMjere(test.sifra=5, test.sifra=6)` i transakciju T_2 `prikaziSumu(test.sifra=5, test.sifra=6)`. Povijest mora biti takva da manifestira problem nekonzistentne analize. Dokazati da H nije CSR.

b) Načiniti tablicu sličnu onoj u 2. zadatku koja će opisivati redoslijed SQL naredbi za T_1 i T_2, koji je konzistentan s poviješću H.

c) Izvršavanjem SQL naredbi utvrditi stvarni (producirani) redoslijed izvršavanja, nacrtati graf povijesti H_P koju je sustav producirao, dokazati da H_P jest CSR.

Tijekom nadziranih provjera/vježbi očekuju se sljedeća znanja i vještine:

- Objasniti vlastita rješenja. Riješiti sličan zadatak, testirati i objasniti rezultate.