Napredne baze podataka

Zaključavanje podataka

Tehničko veleučilište u Zagrebu
Informatičko-računarski odjel

Sadržaj predavanja

- Paralelni pristup podacima
- Zaključavanje podataka
- Vrste ključeva
- Granulacija zaključavanja
- Razina izolacije

•SQL naredbe

- CREATE TABLE racun (br_racun INT, saldo DECIMAL (8,2),UNIQUE KEY br_racun (br_racun));
- INSERT INTO racun VALUES (1, 400.00);
- INSERT INTO racun VALUES (2, 300.00);
- INSERT INTO racun VALUES (3, 89.00);

- SELECT * FROM racun WHERE br_racun = 1;
  - Rezultat je 1 n-torka

	br_racun	saldo
	1	400.00

- SELECT * FROM racun;
  - Rezultat je skup n-torki

	br_racun	saldo
	1	400.00
	2	300.00
	3	89.00

•Djelatni skup n-torki i pripadna kazaljka (Cursor)

- u SQL-u (DBACCESS) djelatni se skup stvara implicitno
  - rezultat sql upita ispisuje se na zaslonu / u datoteci / na pisaču
- ako je SQL naredba sastavni dio nekog programa pisanog u nekom drugom jeziku (C, 4GL, SPL), djelatni skup eksplicitno se definira:
  - pripadna kazaljka - CURSOR omogućuje obradu pojedinih n-torki

```
DELIMITER //
CREATE PROCEDURE pom ( )
BEGIN
      DECLARE pomBr INT;
      DECLARE kraj BOOL;
      DECLARE kur CURSOR FOR
         SELECT  br_racun FROM racun WHERE saldo>100;
      DECLARE CONTINUE HANDLER
         FOR NOT FOUND SET kraj=FALSE;
      OPEN kur;
      SET kraj=TRUE;
      Petlja: LOOP
            FETCH kur INTO pomBr;
            IF NOT kraj THEN
                  LEAVE petlja;
            END IF;
            UPDATE racun SET saldo = saldo * 1.1 ;
      END LOOP;
      CLOSE kur;
END;
//
DELIMITER ;
```

	br_racun	saldo
	1	400.00
	2	300.00
	3	89.00

```
SELECT * FROM racun;
CALL pom();
SELECT * FROM racun;
```

	br_racun	saldo
	1	440.00
	2	330.00
	3	89.00
*	(NULL)	(NULL)

Kontrola paralelnog pristupa

Pregled

- ne dozvoliti istovremenu izmjenu podatka
  - **izgubljene izmjene (lost update)** - posljednji pobjeđuje
- ne dozvoliti čitanje podataka koje netko drugi mijenja
  - **prljavo čitanje (dirty read)**
  - **neponovljivo čitanje (nonrepeatable read)**
  - **sablasti (phantoms)**
- rješenja:
  - zaključavanje podataka
  - skup logički povezanih izmjena - transakcija
  - kontradiktorni zahtjevi
    - zaštita pristupa
    - visok stupanj dostupnosti
  - potpuni zastoj
    - detekcija, predviđanje i prevencija potpunog zastoja
  - vremensko označavanje

Primjer: Izgubljene izmjene

Rezervacija zrakoplovnih karata

- Prodavač A pročita
  - Broj slobodnih mjesta: 20
- Prodavač B pročita
  - Broj slobodnih mjesta: 20
- Prodavač A rezervira 3 mjesta
  - Sustav zapiše broj slobodnih mjesta: 17
- Prodavač B rezervira 2 mjesta
  - Sustav zapiše broj slobodnih mjesta: 18

- Kraj -> u avionu ima 18 slobodnih mjesta!?!

•Primjer: Prljavo čitanje

Program A:

Transakcija A

…

INSERT INTO racun VALUES(4, 320.00);

…

Poništavanje efekata transakcije A (ROLLBACK)

	br_racun	saldo
	1	440.00
	2	330.00
	3	89.00
*	(NULL)	(NULL)

Program B:

SELECT * FROM racun;

	br_racun	saldo
	1	440.00
	2	330.00
	3	89.00
	4	320.00
*	(NULL)	(NULL)

n-torka koja nikada nije postojala!?

	br_racun	saldo
	1	440.00
	2	330.00
	3	89.00
*	(NULL)	(NULL)

SELECT * FROM racun;

•Primjer: Neponovljivo čitanje

Transakcija T1

- SELECT saldo FROM racun WHERE br_racun=2;

Rezultat=400.00

Transakcija T2

- UPDATE racun SET saldo=saldo*1.1 WHERE br_racun=2;

- SELECT saldo FROM racun WHERE br_racun=2;

Rezultat=440.00!!!

•Primjer: Sablasne n-torke (phantom rows)

Transakcija T1

- SELECT COUNT(*) FROM racun WHERE saldo>100;

Rezultat=2

Transakcija T2

- INSERT INTO racun VALUES(4,500.00)

- SELECT COUNT(*) FROM racun WHERE saldo>100;

Rezultat=3!!!

Zaključavanje podataka (locking)

- transakcija može zaključati podatak (podatke)
  - sprječava druge transakcije da pristupe podatku dok ga ona ne otključa
- podaci koji su se mijenjali tijekom transakcije ostaju zaključani do kraja transakcije
- **locking manager** (dio DBMS-a) zaključava zapise i upravlja slučajevima kad postoji više zahtjeva za zaključavanjem istog podatka

Primjer: Izgubljene izmjene (rješenje)

Rezervacija zrakoplovnih karata

- **Prodavač A zaključa tablicu za sebe**
  - Prodavač A pročita
    - Broj slobodnih mjesta: 20
  - Prodavač B ne može pročitati !!
  - Prodavač A rezervira 3 mjesta
  - Sustav zapiše broj slobodnih mjesta: 17
- Prodavač A otpusti tablicu (otključa)
- **Prodavač B zaključa tablicu za sebe**
  - Prodavač B pročita
    - Broj slobodnih mjesta: 17
  - Prodavač B rezervira 2 mjesta
  - Sustav zapiše broj slobodnih mjesta: 15
- Prodavač B otpusti tablicu (otključa)

- Kraj -> u avionu ima 15 slobodnih mjesta !

•Zaključavanje

- DBMS treba omogućiti paralelno izvođenje u što je moguće većoj mjeri
- efekt transakcija koje se obavljaju paralelno mora biti isti kao da su se obavljale jedna iza druge – serijabilnost (serializability)

Program 1

Zaključaj A
Pročitaj A
A = A - 10
Zapiši A
Otključaj A
Zaključaj B
Pročitaj B
B = B + 10
Zapiši B
Otključaj B

Program 2

Zaključaj B
Pročitaj B
B = B - 20
Zapiši B
Otključaj B
Zaključaj C
Pročitaj C
C = C + 20
Zapiši C
Otključaj C

Protokol dvofaznog zaključavanja

Two-phase locking protocol (2PL)

- serijabilnost je osigurana ako sve transakcije poštuju **protokol dvofaznog zaključavanja**
  - prije obavljanja operacije nad objektom (npr. n-torkom iz baze), transakcija mora za taj objekt zatražiti ključ
  - nakon otpuštanja ključa transakcija više ne smije zatražiti nikakav ključ

- transakcije koje poštuju 2PL protokol imaju 2 faze
  - **Fazu pribavljanja ključeva**

    (faza rasta - growing phase)
  - **Fazu otpuštanja ključeva**

    (fazu sužavanja - shrinking phase)
    - faza otpuštanja ključeva najčešće je izvedena kroz jednu operaciju (COMMIT ili ROLLBACK na kraju transakcije)

Vrste zaključavanja

- **KLJUČ ZA PISANJE/IZMJENU - WRITE LOCK**
  - **EXCLUSIVE LOCK** (ekskluzivno zaključavanje)
  - transakcija T1 zaključa objekt za pisanje (da bi mogla pisati)
  - ni jedna druga transakcija ne može zaključati objekt sve dok ga T1 ne otključa
  - svaka operacija izmjene podataka postavlja ključ za pisanje!!!
- **KLJUČ ZA ČITANJE - READ LOCK**
  - **SHARED LOCK** (zajedničko zaključavanje)
  - transakcija T1 zaključa objekt za čitanje (da bi mogla čitati)
  - bilo koja druga transakcija može ga zaključati za čitanje
  - ni jedna transakcija ne može ga zaključati za pisanje

Zaključavanje

- **UNAPRIJEDIVI KLJUČ – PROMOTABLE LOCK**
  - izražava namjeru izmjene
    - na početku se postavlja ključ za čitanje
    - bilo koja druga transakcija također može postaviti ključ za čitanje
    - ni jedna druga transakcija ne može postaviti ključ za pisanje
  - prije izmjene podataka ključ se unaprjeđuje u ključ za pisanje
  - omogućuje DBMS-u da predviđa buduće akcije
  - unaprijedivi ključ za sada ne postoji u MySQL bazi podataka

•Vrste ključeva - pravila

Primjer

Proces 1 postavio je na objekt ključ:

	Read	**Promotable**	**Write**	**Nije zaključano**
Read	OK	OK	Greška	OK
Promotable	OK	Greška	Greška	OK
Write	Greška	Greška	Greška	OK

Proces 2 pokušava postaviti ključ

•Nepotpuni zastoj (Live lock)

Transakcija 1	Transakcija 2	Transakcija 3	Transakcija 4	…
Zaključaj A				
…….	Zaključaj A			
…….				
Otključaj A				
		Zaključaj A		
		……		
		……		
		Otključaj A	Zaključaj A	
			….	
			Otključaj A	

- moguće je da Transakcija 2 čeka zauvijek, iako je mnogo puta imala priliku zaključati podatak A
- rješenje -> strategija **FIRST-COME-FIRST-SERVED**

•Potpuni zastoj (Deadlock)

- **izbjegavanje potpunog zastoja:**
  - transakcija zatraži sva zaključavanja odjednom (npr. na početku)
    - zaključa sve ili ništa!
  - zahtijeva se da transakcije zaključavaju podatke u nekom određenom poretku

Izbjegavanje potpunog zastoja

- **timeout-based deadlock prevention**
  - transakcija čeka na dobivanje ključa
  - ako ga ne dobije tijekom zadanog vremenskog odsječka, menadžer transakcija pretpostavlja da se ključ ne može postaviti upravo zbog pojave potpunog zastoja
  - poništava transakciju koja čeka na dobivanje ključa

- moguće je da će se poništiti transakcije koje zapravo ne sudjeluju u potpunom zastoju

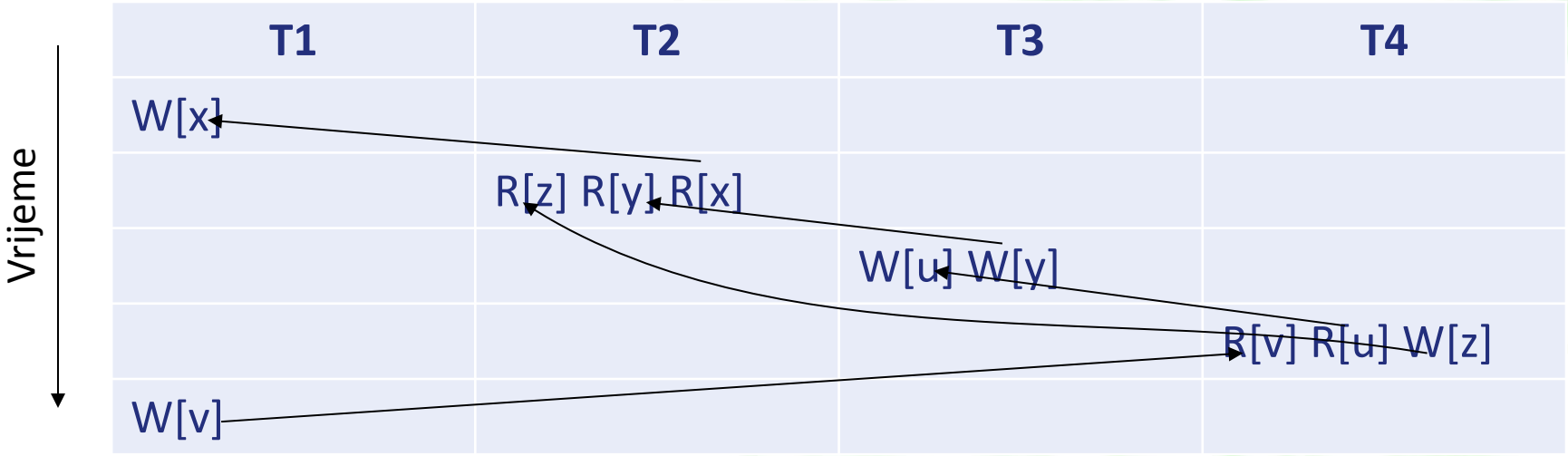
Detekcija potpunog zastoja

- Menadžer transakcija održava **wait-for graf**
  - usmjereni graf čiji su čvorovi označeni identifikatorima transakcija (Ti, Tj, Tk, ...)
  - **luk usmjeren od čvora Ti prema čvoru Tj povlači se ako i samo ako transakcija Ti čeka na postavljanje ključa zbog nekog ključa kojeg je postavila transakcija Tj**

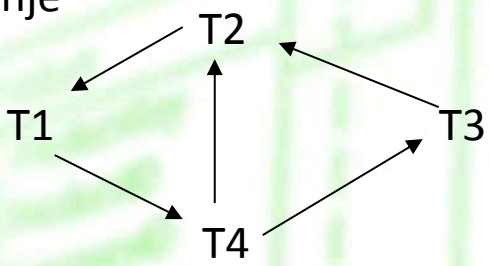
- Potpuni zastoj detektira se otkrivanjem petlji u wait-for grafu

- Petlja se iz grafa uklanja izbacivanjem jednog ili više čvorova (Ti, Tj, Tk,...) i njima pripadnih lukova, uz istovremeno poništavanje pripadnih transakcija Ti, Tj, Tk, ...

•Primjer – wait-for graf



W[x] – zaključavanje zapisa x za pisanje
R[x] – zaključavanje zapisa x za čitanje

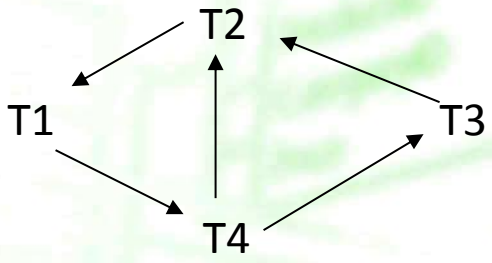


Detektirane tri petlje:
T1,T2,T4
T2,T3,T4
T1,T2,T3,T4

Primjer – wait-for graf

- ispitivanje postoje li petlje u wait-for grafu provodi se
  - permanentno (kod dodavanja svakog novog luka u graf) ili
  - u određenim vremenskim razmacima

- u slučaju da se dogodi potpuni zastoj:
  - barem jedna transakcija mora se prekinuti -> poništavaju se njezini efekti
  - poništavanjem T3 eliminirala bi se samo jedna petlja
  - poništavanjem bilo koje od preostalih transakcija eliminirale bi se obje petlje u grafu



Granulacija zaključavanja

- Određuje se veličina (razina) koja će biti zaključana
  - Baza podataka
  - Tablica (relacija)
  - Redak (n-torka)
  - Indeks
- Istodobnost Vs. smanjenje performansi

Granulacija zaključavanja

Baza podataka

- Zaključavanje baze podataka (DATABASE)
  - Stavljanje ključa:

    ```
    FLUSH TABLES WITH READ LOCK
    ```

  - Skidanje ključa:

    ```
    UNLOCK TABLES
    ```

- Koristi se uglavnom za stvaranje backupa (budući da zaustavlja trenutni rad baze)
- Koristi globalno zaključavanje
- Početak transakcije neće otpustiti globalni ključ (iako implicitno poziva UNLOCK TABLES)

Granulacija zaključavanja

Tablica

- Zaključavanje relacija (datoteka, tablica) - RELATION, TABLE
  - Stavljanje ključa:

```
LOCK TABLES ime_tablice READ |
  [LOW PRIORITY] WRITE
```

  - Skidanje ključa:

```
UNLOCK TABLES
```

- Za izvršavanje potrebna privilegija za LOCK TABLES i SELECT
- LOCK je moguće postaviti na privremenu tablicu ali ga engine ignorira
  - Privremena tablica vrijedi samo u sesiji -> prema tome lock nije potreban

Granulacija zaključavanja

N-torke

- Zaključavanje n-torke (redak) – ROW
  - Stavljanje ključa:

    `SELECT … WHERE … LOCK IN SHARE MODE;`

    `SELECT … WHERE … FOR UPDATE;`
  - Skidanje ključa:

    po završetku transakcije
- SHARE MODE
  - Svim ostalim transakcijama dozvoljeno je čitati , ali ne smiju mijenjati podatke
- FOR UPDATE
  - Ekskluzivni lock, sadržaj n-torki bit će promijenjen, nije dozvoljeno ni čitanje drugim procesima.

Primjer:

```
SELECT * FROM parent WHERE NAME = 'Jones'
  LOCK IN SHARE MODE;
SELECT counter_field FROM child_codes FOR UPDATE;
UPDATE child_codes SET counter_field= counter_field + 1;
```

Granulacija zaključavanja

Indeks

- Zaključavanje indeksa - INDEX, KEY
  - pod kontrolom SUBP
    - Zaključava pojedine elemente s ciljem sprječavanja sablasnih n-torki
    - npr. čuvanje “mjesta” za ključ čiji je zapis obrisan - za slučaj poništavanja transakcije

Razina izolacije (Isolation level)

- određuje način pristupa podacima u višekorisničkom okruženju

1. **prljavo čitanje - DIRTY READ, READ UNCOMMITTED**
   - čitaju se svi traženi podaci bez zaključavanja i bez provjere jesu li možda zaključani!
   - mogu se pojaviti n-torke koje stvarno nikada nisu postojale u bazi podataka!

2. **čitanje potvrđenih n-torki – READ COMMITTED**
   - Proces koji čita **provjerava** je li podatak koji se trenutno čita već zaključan za pisanje - podaci se ne zaključavaju!

Razina izolacije (Isolation level)

3. **ponovljivo čitanje - REPEATABLE READ**
   - dohvat n-torke iz djelatnog skupa uzrokuje zaključavanje (za čitanje) odgovarajuće n-torke u bazi podataka
   - dohvatom nove n-torke prethodna ostaje zaključana
   - kod ponovnog dohvata istog djelatnog skupa unutar iste transakcije, prethodno zaključane n-torke i dalje su zaključane
   - zapisi/stranice ostaju zaključane do kraja transakcije!

4. **ponovljivo čitanje – SERIALIZABLE**
   - isto kao i REPEATABLE READ
   - ostao je radi kompatibilnosti sa InnoDB mehanizmom

Razina izolacije - kako se postavlja

- Može se postaviti
  - globalno (za poslužitelj) ili
  - za sesiju u kojoj trenutno radimo
- Definira se na početku transakcije (nakon BEGIN WORK)

```
SET [GLOBAL | SESSION] TRANSACTION ISOLATION LEVEL
  {
       READ UNCOMMITTED
     | READ COMMITTED
     | REPEATABLE READ
     | SERIALIZABLE
   }
```

•Prljavo čitanje (Read uncommited)

Program A

- Transakcija A
- …

  INSERT INTO racun VALUES(4,320.00);

- …
- Poništavanje efekata transakcije A

	br_racun	saldo
	1	440.00
	2	330.00
	3	89.00
*	(NULL)	(NULL)

	br_racun	saldo
	1	440.00
	2	330.00
	3	89.00
	4	320.00
*	(NULL)	(NULL)

	br_racun	saldo
	1	440.00
	2	330.00
	3	89.00
*	(NULL)	(NULL)

Program B:

- SET TRANSACTION ISOLATION LEVEL **READ UNCOMMITED**;
- SELECT * FROM racun;
- N-torka koja nikada nije postojala ?
- SELECT * FROM racun;

•Čisto čitanje (Read commited)

Program A

- Transakcija A
- …

  INSERT INTO racun
  VALUES(4,320.00);

- …
- Poništavanje efekata transakcije A

	br_racun	saldo
	1	440.00
	2	330.00
	3	89.00
*	(NULL)	(NULL)

	br_racun	saldo
	1	440.00
	2	330.00
	3	89.00
*	(NULL)	(NULL)

Program B

SET TRANSACTION
ISOLATION LEVEL
READ COMMITED;

SELECT * FROM racun;

STOP: zapis zaključan!

SELECT * FROM racun;

Problemi modova izolacije

- READ UNCOMMITTED
  - n-torke koje nikad stvarno nisu postojale u bazi podataka – dirty read problem
  - neponovljivo čitanje
  - sablasne n-torke
- READ COMMITTED
  - neponovljivo čitanje
  - sablasne n-torke
- REPEATABLE READ
  - sablasne n-torke
- SERIALIZABLE
  - sablasne n-torke

Sprječavanje pojavljivanja sablasnih n-torki

- Protokol zaključavanja indeksa (index locking protocol)
  - Svaka relacija mora imati barem jedan indeks
  - Transakcija Ti može pristupati n-torkama relacije isključivo preko jednog ili više indeksa relacije
  - Ako transakcija Ti pregledava podatke – mora postaviti dijeljeni ključ na sve indeksne listove kojima pristupa
  - Transakcija Ti ne može dodavati, mijenjati ili brisati n-torku ti u relaciji r bez izmjene svih indeksa na r
  - Transakcija mora pribaviti ekskluzivne ključeve na svim indeksnim listovima na koje utječe unos, izmjena ili brisanje
  - Moraju biti primijenjena pravila protokola dvofaznog zaključavanja