Veri Tabanı Sistemleri

Eşzamanlı Çalışma

H. Turgut Uyar Şule Öğüdücü

2002-2016

License

Hareket Yönetimi

- birden fazla işlemin topluca yapılması gerekebilir
- bir işlemin yapılıp diğerlerinin yapılmaması tutarsızlık yaratabilir
- hareket: bir işin mantıksal bir birimi
- birden fazla işlemin topluca yapılması garanti edilemez
- en azından değişikliklerden önceki duruma dönülebilmeli

Hareket Örneği

- bir banka hesabından diğerine para aktarma

```
UPDATE ACCOUNTS SET BALANCE = BALANCE - 100
  WHERE ACCOUNTID = 123

UPDATE ACCOUNTS SET BALANCE = BALANCE + 100
  WHERE ACCOUNTID = 456
```

Hareket Özellikleri

- bölünmezlik: ya tam yapılır, ya hiç yapılmaz
- tutarlılık: bir tutarlı durumdan diğer bir tutarlı duruma geçiş
- yalıtım: sona ermemiş bir hareketin işlemlerinin diğer hareketleri etkileyip etkilemediği
- kalıcılık: bir hareket sonlandırıldıktan sonra sistem çökse de verilerin zarar görmemesi

SQL Hareketleri

- başlatma

  `BEGIN [ WORK | TRANSACTION ]`

- sonlandırma

  `COMMIT [ WORK | TRANSACTION ]`

- vazgeçme

  `ROLLBACK [ WORK | TRANSACTION ]`

Hareket Örneği

```
BEGIN TRANSACTION
ON ERROR GOTO UNDO
UPDATE ACCOUNTS SET BALANCE = BALANCE - 100
  WHERE (ACCOUNTID = 123)
UPDATE ACCOUNTS SET BALANCE = BALANCE + 100
  WHERE (ACCOUNTID = 456)
COMMIT
...

UNDO:
  ROLLBACK
```


Sistemin Düzeltilmesi

- bir hareket sürerken sistemin çöktüğünü düşünün
- bellek tamponlarındaki veriler diske yazılmamış durumda
- kalıcılık nasıl sağlanacak?
- veri, sistemde başka yerde yazılı verilerden türetilebilmeli

Günlük

- günlük her işlemden etkilenen her çoklunun işlemden önceki ve sonraki değerlerini tutar
- günlüğe önceden yazma kuralı: hareket sonlanmadan önce günlük fiziksel ortama yazılmalı
- günlük kayıtlarına erişim işlemin doğası gereği ardışıl

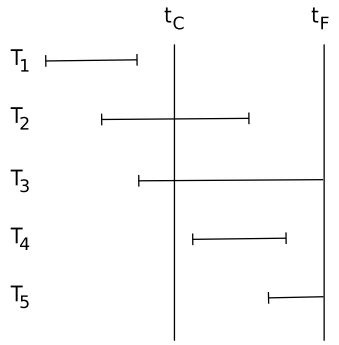
Denetim Noktaları

- belli aralıklarla günlükte denetim noktaları oluşturulur
- bellek tamponlarındaki veriler fiziksel ortama yazılır
- denetim noktası günlüğe not edilir
- o an sürmekte olan hareketler not edilir

Düzeltme Listeleri

- aksaklıktan sonra hangi hareketler geri alınacak, hangileri sonlandırılacak?
- iki liste oluştur: *geri alınacaklar* (G), *yinelenecekler* (Y)
- t_C: günlükte kayıtlı son denetim noktası
- t_C anında etkin olan hareketleri G'ye ekle
- t_C'den başlayarak kayıtları ileri doğru tara
- başlayan bir hareketle karşılaşırsan G'ye ekle
- biten bir hareketle karşılaşırsan Y'ye geçir

Düzeltme Örneği



- t_C: $G = [T_2, T_3]$ $Y = []$
- T_4 başladı: $G = [T_2, T_3, T_4]$ $Y = []$
- T_2 bitti: $G = [T_3, T_4]$ $Y = [T_2]$
- T_5 başladı: $G = [T_3, T_4, T_5]$ $Y = [T_2]$
- T_4 bitti: $G = [T_3, T_5]$ $Y = [T_2, T_4]$

Düzeltme Süreci

- kayıtları günlük sonundan geriye doğru tara
- G'deki hareketlerin yaptıkları değişiklikleri geri al
- kayıtları ileriye doğru tara
- Y'deki hareketlerin yaptıkları değişiklikleri yinele

İki Aşamalı Sonlandırma

- farklı kaynak yöneticileri var
- geri alma - sonlandırma sistemleri ayrı
- etkilenecek veriler farklı kaynak yöneticilerinde
- ya hepsinde birden sonlandırılacak
  ya da hepsinde birden geri alınacak
- eşgüdüm sağlayıcı

Protokol

- eşgüdüm sağlayıcı → katılımcılar:
  hareketle ilgili bütün verilerin kayıtlarını kalıcı ortama yaz
- eşgüdüm sağlayıcı → katılımcılar:
  hareketi başlat ve sonucu bildir
- bütün katılımcılar başarılı: başarı
- diğer durumda: başarısız
- başarı ise: eşgüdüm sağlayıcı → katılımcılar: sonlandır
- başarısız ise: eşgüdüm sağlayıcı → katılımcılar: vazgeç

Kaynaklar

Okunacak: Date

- Chapter 15: Recovery

Eşzamanlı Çalışma

- eşzamanlı çalışan hareketler nedeniyle çıkabilecek sorunlar:
- yitirilen güncelleme
- kesinleşmemiş veriye bağımlılık
- tutarsız çözümleme

Yitirilen Güncelleme

Hareket A	Hareket B
...	...
RETRIEVE p	...
...	...
...	RETRIEVE p
...	...
UPDATE p	...
...	...
...	UPDATE p
...	...

Kesinleşmemiş Veriye Bağımlılık

Hareket A	Hareket B
...	...
...	UPDATE p
...	...
RETRIEVE p	...
...	...
...	ROLLBACK
...	

Tutarsız Çözümleme

- hesap toplamı: acc1=40, acc2=50, acc3=30

Hareket A	Hareket B
...	...
RETRIEVE acc1 (40)	...
RETRIEVE acc2 (90)	...
...	...
...	UPDATE acc3 (30 $\rightarrow$ 20)
...	UPDATE acc1 (40 $\rightarrow$ 50)
...	COMMIT
...	...
RETRIEVE acc3 (110)	
...	

Conflicts

- A okuyor, B okuyor: sorun yok
- A okuyor, B yazıyor: yinelenemez okuma (tutarsız çözümleme)
- A yazıyor, B okuyor: kirli okuma (kesinleşmemiş veriye bağımlılık)
- A yazıyor, B yazıyor: kirli yazma (yitirilen güncelleme)

Locking

- hareketler üzerinde işlem yapacakları çokluları kilitlesinler
- okuma kilidi (S)
- yazma kilidi (X)

Kilit İstekleri

kilit tipi uyumluluk matrisi

	X	S	-
X	H	H	E
S	H	E	E

- okuma kildi varsa: sadece okuma kilidi istekleri kabul edilir
- yazma kilidi varsa: bütün kilit istekleri reddedilir

Kilitleme Protokolü

- hareket, yapmak istediği işleme göre kilit isteğinde bulunur
- okuma kilidi varsa yazma kilidine çevrilmesi
- istek yerine getirilemiyorsa beklemeye başlar
- diğer hareket kilidi bırakınca devam eder
- sonsuz bekleme

Yitirilen Güncelleme

Hareket A	Hareket B
...	...
RETRIEVE p (S+)	...
...	...
...	RETRIEVE p (S+)
...	...
UPDATE p (X-)	...
bekle	...
bekle	UPDATE p (X-)
bekle	bekle

Kesinleşmemiş Veriye Bağımlılık

Hareket A	Hareket B
...	...
...	UPDATE p (X+)
...	...
RETRIEVE p (S-)	...
bekle	...
bekle	ROLLBACK
RETRIEVE p (S+)	
...	

Tutarsız Çözümleme

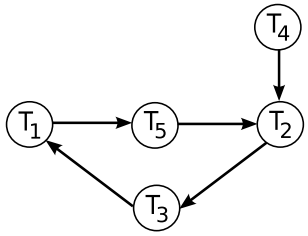
Hareket A	Hareket B
...	...
RETRIEVE acc1 (S+)	...
RETRIEVE acc2 (S+)	...
...	...
...	UPDATE acc3 (X+)
...	UPDATE acc1 (X-)
...	bekle
RETRIEVE acc3 (S-)	bekle
bekle	bekle

Ölümcül Kilitlenme

- ölümcül kilitlenme: hareketlerin birbirlerinin kilitleri bırakmalarını beklemesi
- neredeyse her zaman iki hareket arasında
- farketmek ve çözmek
- önlemek

Ölümcül Kilitlenmenin Çözülmesi

örnek



- bekleme grafı
- bir kurban seç ve öldür

Ölümcül Kilitlenmenin Önlenmesi

- her hareketin başlama zamanı mührü var
- A hareketinin kilit isteği
  B hareketinin tuttuğu bir kilitle çelişiyorsa:
- bekle-öl: A, B'den yaşlıysa bekler, gençse ölür
  A geri alınıp yeniden başlatılır
- yarala-bekle: A, B'den gençse bekler, yaşlıysa B'yi yaralar
  B geri alınıp yeniden başlatılır
- yeniden başlatılan hareketin zaman mührü değiştirilmez

Kilit Komutları

- okuma kilidi

  ```
  SELECT query FOR SHARE
  ```

- yazma kilidi

  ```
  SELECT query FOR UPDATE
  ```


Yalıtım Düzeyleri

- yalıtım azaltılırsa eşzamanlılık artırılabilir
- değişik yalıtım düzeyleri:

- serileştirilebilir
- yinelenebilir okuma
- sonlandırılanları okuyabilme
- sonlandırılmayanları okuyabilme

Serileştirilebilirlik

- *seri çalıştırma*:
  hareketlerin biri bitmeden diğeri başlamıyor
- serileştirilebilir: eşzamanlı çalışmanın sonucu
  her zaman seri çalıştırmalardan birinin sonucu ile aynı

örnek

- $x = 10$
- A hareketi: $x = x + 1$
- B hareketi: $x = 2 * x$
- önce A, sonra B: $x = 22$
- önce B, sonra A: $x = 21$

İki Aşamalı Kilitleme

- iki aşamalı kilitleme:
  herhangi bir kilit bırakıldıktan sonra
  yeni kilit isteğinde bulunulmaz
- genişleme aşaması: alınan kilit sayısı artıyor
- daralma aşaması: alınan kilit sayısı azalıyor
- iki aşamalı sıkı kilitleme:
  bütün kilitler hareketin sonunda bırakılır
- *Bütün hareketler iki aşamalı kilitleme protokolüne uyarsa bütün eşzamanlı çalıştırmalar serileştirilebilir.*

Sonlandırılanları Okuyabilme

- read commited: yalnızca yazma kilitleri hareket sonuna kadar tutulur

Hareket A	Hareket B
...	...
RETRIEVE p (S+)	...
...	...
kilidi bırak	...
...	...
...	UPDATE p (X+)
...	COMMIT
RETRIEVE p (S+)	

Hayaletler

- hayalet: sorgu yeniden çalıştırıldığında yeni çoklular ortaya çıkıyor

örnek

- A hareketi bir müşterinin hesaplarının ortalamasını hesaplıyor:
  $\frac{100+100+100}{3} = 100$
- B hareketi aynı müşteriye (200) birimlik yeni bir hesap yaratıyor
- A hareketi hesabı yeniden yapıyor:
  $\frac{100+100+100+200}{4} = 125$

Yalıtım Düzeyleri

- bir yalıtım düzeyi belirleme

```
SET TRANSACTION ISOLATION LEVEL
  [ SERIALIZABLE | REPEATABLE READ |
    READ COMMITTED | READ UNCOMMITTED ]
```


Yalıtım Düzeyi Sorunları

yalıtım düzeyi	kirli okuma	yinelemeyen okuma	hayalet
READ UNCOMMITTED	E	E	E
READ COMMITTED	H	E	E
REPEATABLE READ	H	H	E
SERIALIZABLE	H	H	H

Kilitleme Birimi

- kilitleme çoklu değil bağıntı değişkeni biriminde yapılabilir
- hatta veri tabanı biriminde
- birim: kilitlenen birim
- birim genişledikçe eşzamanlılık azalır
- çoklular üzerinde alınmış kilitlerin bulunması zor
  $\rightarrow$ önce bağıntı değişkeni düzeyinde niyet kilitleri alınsın

Niyet Kilitleri

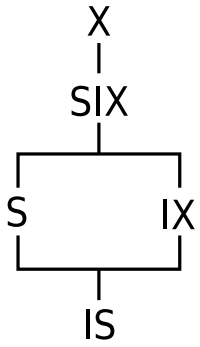
- Parçayı Okuma (IS):
  hareket bazı çokluları okumaya niyetleniyor
- Parçaya Yazma (IX):
  IS + hareket bazı çoklulara yazmaya niyetleniyor
- Bütünü Okuma (S):
  bağıntıda eşzamanlı okuyucular olabilir ama yazıcılar olmamalı
- Bütünü Okuma + Parçaya Yazma (SIX):
  S + IX
- Bütüne Yazma (X):
  bağıntıda hiçbir eşzamanlı çalışma olmamalı

Kilit İstekleri

kilit uyumluluk matrisi

	X	SIX	IX	S	IS	-
X	H	H	H	H	H	E
SIX	H	H	H	H	E	E
IX	H	H	E	H	E	E
S	H	H	H	E	E	E
IS	H	E	E	E	E	E

Kilit Öncelikleri



- çoklu üzerinde okuma kilidi için bağıntı üzerinde en az IS
- çoklu üzerinde yazma kilidi için bağıntı üzerinde en az IX

Kilitleme Komutları

- bir tabloyu kilitleme

```
LOCK [ TABLE ] table_name
     [ IN lock_mode MODE ]
```

- kilit kipleri:
  - ACCESS SHARE
  - ROW SHARE
  - ROW EXCLUSIVE
  - SHARE UPDATE EXCLUSIVE
  - SHARE
  - SHARE ROW EXCLUSIVE
  - EXCLUSIVE
  - ACCESS EXCLUSIVE

Kaynaklar

Okunacak: Date

- Chapter 16: Concurrency