Увод у релационе базе података

13а

Саша Малков
Универзитет у Београду
Математички факултет
2023/2024

[PM13]
Увод у РБП

Саша Малков

Тема 10
Трансакције

Основни појмови

- Трансакција
  - целовит посао који се ***или*** обави у потпуности ***или*** не оставља никакве промене
- Потврђивање трансакције (енгл. ***commit***)
  - саопштавање да смо успешно довршили све планирамне промене у оквиру трансакције и да желимо да их трајно запишемо
- Поништавање трансакције (енгл. ***rollback***)
  - саопштавање да нисмо успешно довршили све планиране промене и да желимо да поништимо све до сада извршене промене

Пример трансакције

- Нека је потребно да пребацимо новац са рачуна *A* на рачун *B*:

```
update racun
set stanje = stanje - 1000
where br_racuna = :A
update racun
set stanje = stanje + 1000
where br_racuna = :B
```

- Претходне наредбе морају или да обе буду извршене или да обе буду неизвршене
- Не сме да се догоди да само једна од њих остави траг у систему
- То је пример трансакције

Форма трансакције

- Трансакције имају следећу форму:

```
begin transacion        -- започињање трансакције
...naredba 1...
if error... goto greska
...naredba 2...
if error... goto greska
...
...naredba N...
if error... goto greska
commit        -- потврђивање трансакције
goto kraj

greska:
  rollback      -- поништавање трансакције
kraj:
  ...
```


Наредбе везане за трансакције

- BEGIN TRANSACTION
  - започињање трансакције
  - код већине савремених СУБП се не користи – трансакција започиње првом наредбом која приступа садржају базе података
- COMMIT
  - потврђивање успешно довршене трансакције
  - ако се ова наредба успешно изврши, све промене су трајно записане
- ROLLBACK
  - поништавање извршених делова недовршене трансакције
  - ако се трансакција прекине насилно (испадом из процеса и сл.), последице по трансакцију су као да је извршена ова наредба

Особине трансакција (*ACID*)

- **Атомичност** (енгл. **atomicity**)
  - трансакција је целовита
  - или је цела извршена или ниједан њен део није извршен
- **Конзистентност** (енгл. ***consistency***)
  - трансакција мора да чува конзистетност базе података
  - преводи базу из једног у друго исправно стање
- **Изолованост** (енгл. ***isolation***)
  - трансакције су међусобно изоловане током извршавања
  - једна трансакција не сме да има утицај на ефективан рад друге трансакције
    - тј. може да је успори или привремено заустави, али не утиче на резултат
- **Трајност** (енгл. ***durability***)
  - по успешном потврђивању трансакције су трајно записане
  - чак и пад система не утиче на поузданост записа

Тачке чувања (енгл. *savepoints*)

- Тачка чувања
  - означава неки тренутак током извршавања трансакције
  - не представља потврђивање
  - већ касније извршавање може да се поништи или од почетка или само од те тачке
- Омогућавају да се трансакција подели на делове
- Али и даље се трансакција потврђује само као целина

Имплементација транскција

- Уобичајено се за имплементацију користи комбинација два концепта:
  - Катанци
    - служе као техника за изоловање трансакција
  - Дневник (енгл. *log*)
    - служи за имплементацију атомичности и трајности
- Конзистентност се имплементира од стране програмера, који води рачуна о томе да трансакција преводи једно исправно стање у друго исправно стање базе података
  - при томе помаже атомичност

Проблем изоловања

- Основне врсте проблема при раду са конкурентним трансакцијама, а без решавања изловања, су:
  - **Изгубљене промене**
  - **Читање непотврђених измена**
  - **Непоновљиво читање**
  - **Фантомски редови**
- Основна техника решавања су катанци

Катанци

- Механизам катанаца је уобичајено средство остваривања изолованости транскација
- Основна идеја је да прва трансакција која приступа податку закључа тај податак тако да друге трансакције не могу да га користе док она не заврши посао

Стратегије закључавања

- Називају се и стратегије *изоловања*
- Две основне стратегије су
  - песимистичко закључавање
  - оптимистичко закључавање

Песимистичко закључавање

- Песимистичко закључавање је уобичајено за базе података
  - претпоставља се да ће и други процеси (трансакције) користити податке у исто време
  - сваки податак се пре приступања закључава одговарајућим катанцем
  - катанац се чува до краја трансакције и тако се обезбеђује да нико не може да му приступи на нерегуларан начин
- У пракси је песимистичко закључавање релативно скупо
  - закључавају се сви подаци којима се приступа
- Зато се релаксира увођењем *нивоа изолованости*
  - ...детаљније ускоро...

Оптимистичко закључавање

- Оптимистичко закључавање је уобичајено за дуготрајне трансакције
  - нпр. отвори се формулар и попуњава се или ажурира врло дуго, можда и цео сат или дуже
  - карактеристично је за веб-апликације
- Претпоставља се да други неће користити податке
  - тј. да је вероватноћа истовременог приступа истим подацима врло ниска
- Обично се остварује на следећи начин:
  - при читању се запамти када је податак прочитан
  - када дође време за писање
    - сва писања чине једну *песимистичку* трансаксију
    - за сваки податак на коме почива трансакција се проверава да ли је у међувремену био промењен
    - ако јесте, трансакција се прекида

Оптимистичко закључавање у БП

- Оптимистичко закључавање обично примењују апликације или апликативни развојни оквири
- База података може да подржи оптимистичко закључавање
  - свакој табели се додаје колона са временом последње промене реда
  - или се додаје *токен измена*
  - потенцијално се имплицитно скривају додатне колоне
  - или се додаје функција за читање идентификатора реда

Катанци (2)

- Сваки катанац има:
  - објекат који се закључава
  - трајање
  - врсту катанца

Предмет закључавања

- Објекат може да буде
  - вредност (атрибут)
  - ред табеле
  - страница табеле
  - група страница
  - цела табела
  - простор за табеле
  - индекс
- Величина објекта одређује *грануларност* катанца

Ескалација катанаца

- Величина катанца
  - Катанац може да закључава један ред или више редова
  - Или једну страницу или више страница
- Велики број катанаца може да значајно успори рад
  - због много више провера при раду...
  - зато је број катанаца ограничен (било бројем катанаца било количином меморије која је резервисана за катанце)
- Када се превазиђе допуштен број катанаца долази до *ескалације катанаца*
  - Више мањих катанаца се замењује једним већим
  - На пример, више катанаца на редовима се замењује катанцем на страници
  - Или, више катанаца на страницама се замењује катанцем на табели

Врста катанаца

- Две основне врсте катанаца су
  - ексклузиван катанац (енгл. *exclusive*)
    - поставља се пре операције која мења садржај објекта базе података
    - може да се постави само ако не постоји други катанац на истом објекту
    - док постоји, не може да постави ниједан други катанац на истом објекту
  - дељиви катанац (енгл. *shared*)
    - поставља се пре операције која само чита садржај објекта базе података
    - може да се постави ако не постоји други катанац или ако постоји други (један или више њих) дељиви катанац
    - док постоји, могу да се постављају други дељиви катанци на истом објекту

Врста катанаца (2)

- Додаје се катанац за ажурирање:
  - поставља се пре операције која мења садржај објекта базе података, али иза које се очекује да следи операција мењања истог објекта
  - пре операције мењања мора да се промовише у ексклузиван катанац
  - може да се постави само ако не постоји други катанац или ако постоји неки (један или више) дељиви катанац на истом објекту
  - док постоји, могу да се постављају само други дељиви катанци на истом објекту
    - не могу да постоје истовремено два катанца за ажурирање

Врста катанаца (3)

\ Нови \ Постоји \	S	U	X
S	+	+	-
U	+	-	-
X	-	-	-

Врста катанаца (4)

- У пракси имплементације СУБМ додају још неколико врста катанаца, прилагођених употреби различитих објеката
  - циљ је да се олакша комбиновање операција које не ометају једне друге, а да се при томе не доведе у питање излованост

Трајање катанца

- У идеалном случају катанац траје до завршетка трансакције која га је поставила
- Такав приступ решава све могуће проблеме...
- Али доводи до много узајамног чекања и свеукупног успоравања рада СУБП
  - зато се праве компромиси

Нивои изолованости

- Нивои изолованости одређују у којој мери једна трансакција ради изоловано и независно од других трансакција
  - Уводе се ради подизања ефикасности система
  - Доводе до потенцијалних грешака у раду
    - ако се не користе исправно и пажљиво
- Односе се на начин постављања и трајање дељивих катанаца
  - и њихових посебних врста, попут катанаца за ажурирање
  - не утичу на рад ексклузивних катанаца
    - они се свакако захтевају пре сваке операције мењања и ослобађају тек по окончању трансакције

Нивои изолованости (2)

- Нивои изолованости:
  - **серијализујући** (енгл. *serializable*)
    - у терминима *DB2* "*repeatable read*"
    - потпуно изоловање, као у идеалном случају
  - **поновљиво читање** (енгл. *repeatable reads*)
    - у терминима *DB2* "*read stability*"
    - дељиви катанци се постављају само на редовима који испуњавају услов упита, тј. ефективно се читају и користе
    - може да дође до проблема фантомских редова
  - **читање потврђених** (енгл. *read committed*)
    - у терминима *DB2* "*cursor stability*"
    - додатно, дељиви катанци се ослобађају када се при читању пређе на наредни ред
    - може да дође и до проблема непоновљивог читања
  - **читање непотврђених** (енгл. *read uncommitted*)
    - у терминима *DB2* "*uncommitted read*"
    - дељиви катанци се ни не постављају при читању
    - може да дође и до проблема читања непотврђених измена

Дневник

- Дневник је фајл у коме се серијски записује ток свих промена на бази података
  - за сваку промену се записују
    - ид трансакције
    - време
    - врста и опис операције
    - детаљан садржај захваћених података
  - важне тачке синхронизације захтевају трајно записивање (*buffer flush*)
    - пре свега потврђивање трансакције
- Наредба *commit* је успешно извршена тек након што је њен траг трајно записан у дневнику
  - физичка промена садржаја табеле (записивање измењених страница) се обично одвија асинхроно и не мора да у том тренутку буде довршено

Дневник (2)

- На тај начин дневник остварује и трајност и атомичност
- Додатно, садржај дневника може да се користи и за опоравак базе података
  - тзв. "размотавање дневника"
  - све трансакције које су потврђене (постоји запис у дневнику) а чије промене нису уписане у табеле базе података, понављају се на основу дневника и уписују у табеле

Литература за тему

- Гордана Павловић-Лажетић, **Увод у релационе базе података,** *2.изд. Математички факултет, 1999.*
  - доступно онлајн: *http://poincare.matf.bg.ac.rs/~gordana//urbp-2016.htm*
- Документација за *DB2 11.5*:
  - онлајн:
    - https://www.ibm.com/support/knowledgecenter/SSEPGG_11.5.0
  - ПДФ:
    - https://www.ibm.com/support/pages/node/627743
- Визуализација алгоритама:
  - https://www.cs.usfca.edu/~galles/visualization/Algorithms.html