Adminstracija baza podataka

Transakcije nad bazom podataka

Šta je Transakcija? (1/5)

- Transakcija je vremenski uređen niz nedeljivih radnji nad bazom podataka koje u celini ne remete uslove integrideta.
- Transakcija je osnovna nedeljiva jedinica rada nad bazom podataka koja čini jednu logičku celinu posla.
- Sa aspekta definisanih uslova integriteta transakcija transformiše jedno konzistentno u baze podataka u drugo takođe konzistentno stanje baze podataka.
- Između dva konzistentna stanja dozvoljeno je da se baza podataka nađe i u nekonzistentnom stanju.

Šta je Transakcija? (2/5)

- Bitno je da transakcija bude tako oformljena da iz bilo kojih razloga (npr. otkaz hardvera ili softvera, višekorisnički rad …) ne ostavi trajno bazu podataka u nekonzistentnom stanju.

- Potrebno je obezbediti mehanizam da se transakcija u celini uspešno izvrši (obzirom da je nedeljiva), ili da se, ukoliko dođe do greške, poništе sva njena dejstva i baza podataka vrati u stanje pre početka izvršavanja transakcije.

- Ukoliko dođe do greške u toku izvršavanja transakcije pre njenog kraja, a od početka tra nsakcije je bilo ažuriranja, potrebno je poništiti sva ta ažuriranja i bazu podtataka vratiti u stanje pre početka izvršavanja transakcije.

Šta je Transakcija? (3/5)

- Sa tačke gledišta uslova integriteta transakcija se ponaša ka jedinična radnja, što ne važi za pojedinačne radnje od kojih se transakcija sastoji.
- SUBP koji podržavaju ***transakcionu obradu*** moraju da garantuju da, ukoliko se transakcijom vršilo ažuriranje baze podatakai iz bilo kojih razloga transakcija se nije normalno završila, ponište sva ažuriranja delimično izvršenih transakcija.
- Na taj način transakcija se ili u potpunosti izvršava ili u potpunosti poništavaju njena dejstva.
- Transakciona obrada odvija se pod kontrolom ***administratora transakcija (transaction manager-a)*** kao dela (modula/komponente) SUBP-a čije naredbe COMMIT i ROLLBACK određuju način njegovog funkcionisanja

Šta je Transakcija? (4/5)

- Transakcija prevodi bazu podataka iz jednog konzistentnog stanja u drugo (ne nužno novo) takođe konzistentno stanje baze podataka.
- Transakcije mogu biti jedinice ili sekvence rada izvršene logičkim redom, bilo ručno od strane korisnika ili automatski nekim programom koji upravlja bazom podataka.
- Transakcija je propagiranje jedne ili više promena nad bazom podataka.
- Na primer: Kada vršimo insert, update ili delete vrednosti iz baze, mi vršimo transak ciju nad tabelom.

Šta je Transakcija? (5/5)

- Veoma je važno upravljati transakcijama kako bismo obezbedili integritet podata ka i da bismo mogli da upravljamo greškama.
- Praktično ćemo više SQL upita grupisati i izvršiti ih kao jednu logiču nedeljivu celi nu, odnosno transakciju.

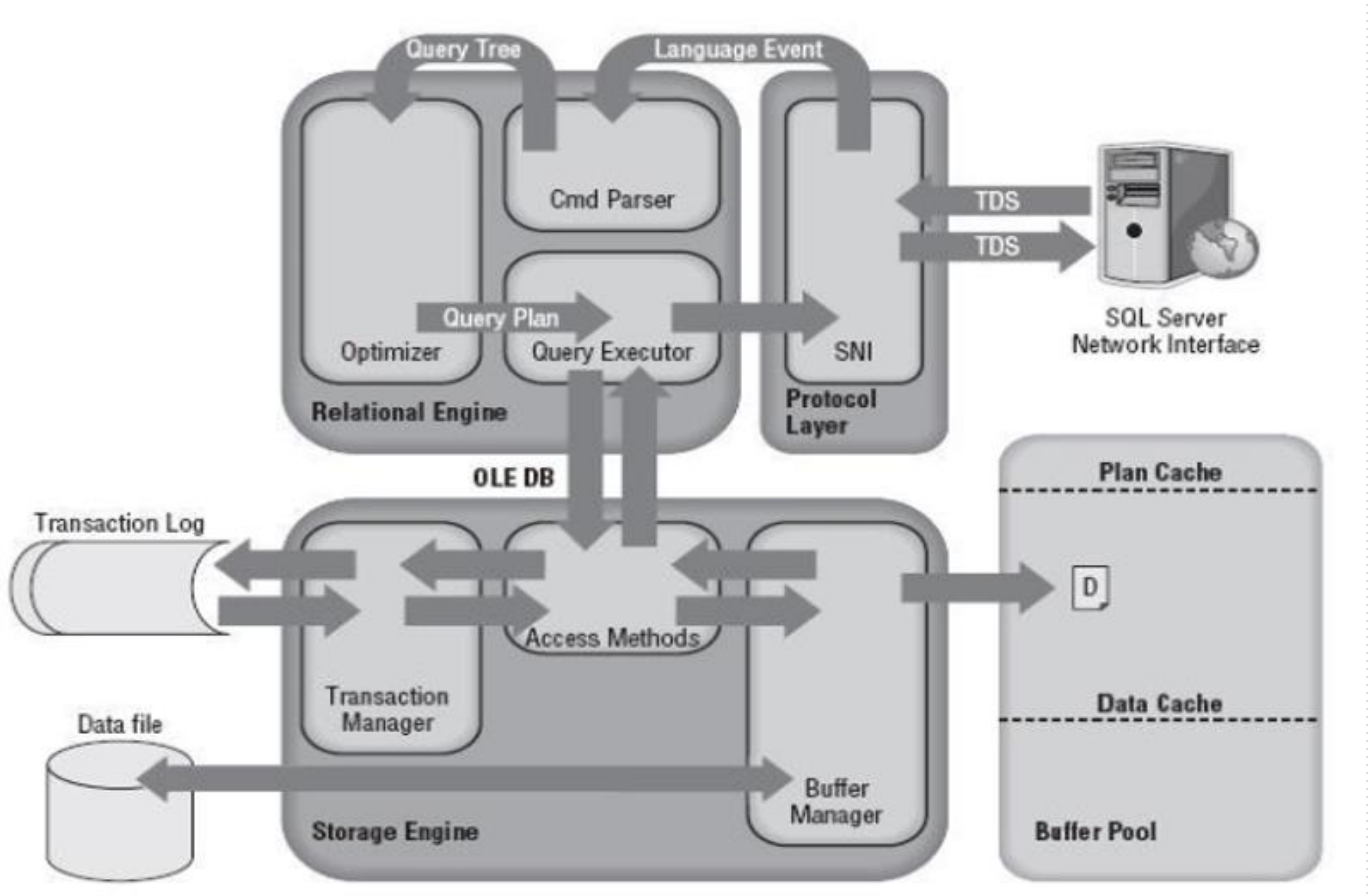
Nekoliko jednostavnih pitanja i odgovora u vezi transakcija

- Ko oformljuje i definiše transakciju?
  - Transakciju nad bazom podataka oformljuje i definiše korisnik koji želi da izvrši neku radnju nad bazom podataka.
  - Korisnik može biti aplikacija, korisnički program ili osoba koja ima pristup bazi podataka.
  - Aplikaciju osmišljava (dizajnira i piše njen programski kod) progamer, tako da on definiše i osmišljava i transakciju nad bazom podataka u okviru svog izvornog programa.
  - Pri tome programer ili administrator baze podataka koristi mehanizme i naredbe koje podržava SUBP, odnosno njegova komponenta ***tranasction manager***.

Šta je Administrator transakcija (*Transaction manager*)?

- Transaction Manager u SUBP-u je softverska komponenta koja upravlja transakcijama nad bazom podataka.
- Ukratko, ***Transaction Manager*** u SUBP-u je ključna komponenta za održavanje integriteta baze podataka, obezbeđuje da se podaci sigurno i pouzdano čuvaju i obrađuju.

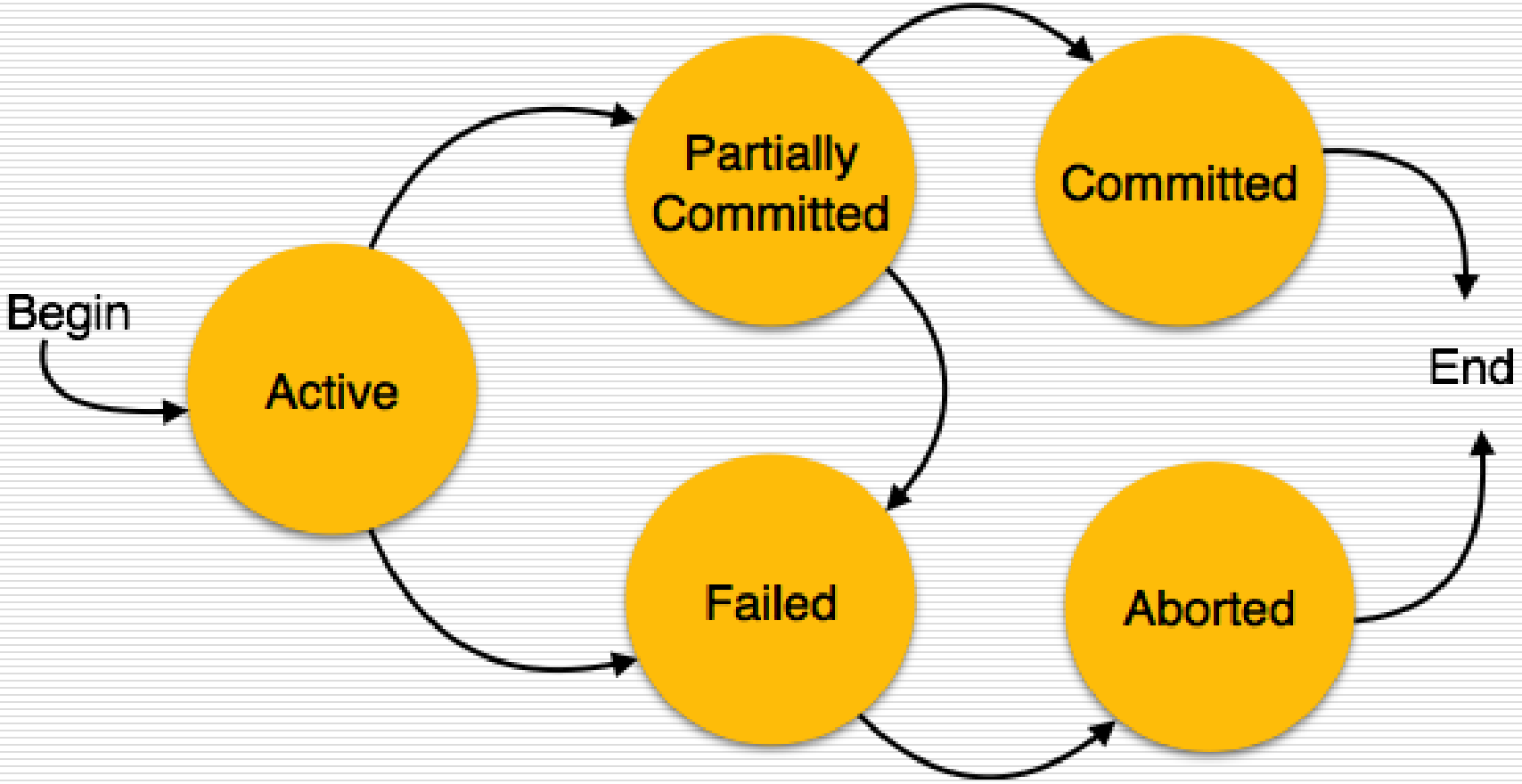
Mesto Administratora transakcija u okviru SUBP-a



Osobine transakcija: *ACID*

- Transakcije imaju 4 standardne osobine, obično opisane akronimom ***ACID***.
- ***Atomicity*** (atomarnost) – obezbeđuje nedjeljivost transakcije, što znači da se sve operacije u transakciji izvršavaju u potpunosti ili se ne izvršavaju uopšte.
- Ako se bilo koja operacija u transakciji ne može izvršiti, sve promene se poništavaju i vraćaju u početno stanje pre početka transakcije (rollback operacija).
- Što znači da će transakcija ili biti izvršena u potpunosti ili će se poništiti sva njena dejstva.
- Ne sme postojati stanje baze u kome je transakcija samo parcijalno izvršena.

Osobine transakcija: ACID



Osobine transakcija: *ACID*

- ***Consistency*** (konzistentnost) – obezbeđuje da će baza pravilno promeniti stanje nakon uspešno izvršene transakcije.
- Nijedna transakcija ne sme ostaviti negativan efekat na podatke koji se nalaze u bazi podataka.
- Ako je baza bila u konzistetnom stanju pre izvršene transakcije, mora ostati u konzistentnom stanju i nakon izvršene transakcije.

Osobine transakcija: ACID

- ***Isolation*** (izolacija) – obezbeđuje da nijedna transakcija neće ugroziti izvrsavanje neke druge transakcije.
- U sistemu baza podataka, gde se jedna ili više transakcija izvršava redom, ili paralelno, izolacija omogućava da svaka transakcija bude izvršena kao da je jedina transakcija u sistemu.

Osobine transakcija: *ACID*

- ***Durability*** (trajnost) – obezbeđuje da će rezultat izvršene transakcije postojati i u slučaju otkaza sistema.
- Baza podataka bi trebalo da bude trajna toliko da zadrži i poslednje napravljene izmene čak i u slučaju otkaza sistema ili ponovnog pokretanja.
- Ako transakcija promeni deo podataka u bazi i ako je izvršena u potpunosti, promene nad bazom podataka će biti trajne čak id da neposredo posle toga dođe do "ispada" bilo kog dela sistema.
- U slučaju da transakcija komituje, ali da baza otkaže pre nego što se podaci upišu na disk, podaci će biti ažurirani čim se sistem vrati u operativno stanje.

Kontrola transakcija

- Postoji nekoliko komandi za upravljanje transakcijama:
- COMMIT – da sačuvamo promene.
- ROLLBACK – da vratimo na prethodno stanje.
- SAVEPOINT – tačka u transakciji na koju možemo da se vratimo bez potrebe za rollback-om cele transakcije
- SET TRANSACTION – postavljamo ime transakciji.

Kontrola transakcija

- Komande za kontrolu transakcija se koriste isključivo za naredbe INSERT, UPDATE i DELETE.
- Ne mogu se koristiti prilikom kreiranja ili brisanja tabele, zato što se ove operacije automatski komituju u bazu podataka.

Commit

- **COMMIT** – komanda koja se koristi da sačuvamo izmene koje je napravila transakcija.
- Commit komanda čuva sve transakcije u bazi, od poslednje Commit ili Rollback naredbe.
- Sintaksa za Commit naredbu:
- COMMIT;

Commit

ID	Name	Age	Salary
1	Milan	25	3000
2	Goran	23	1450
3	Marko	25	2000
4	Petar	27	5000
5	Jovan	29	3150

ID	Name	Age	Salary
2	Goran	23	1450
4	Petar	27	5000
5	Jovan	29	3150

> DELETE FROM CUSTOMERS WHERE AGE = 25;

> COMMIT;

Rollback

- **ROLLBACK** – je komanda kojom poništavamo akcije transakcije koje još uvek nisu sačuvane u bazi podataka.
- Rollback možemo koristiti da poništimo samo one transakcije koje su izvršene nakon poslednjeg commit-a ili rollback-a.
- Sintaksa za Rollback naredbu:
- ROLLBACK;

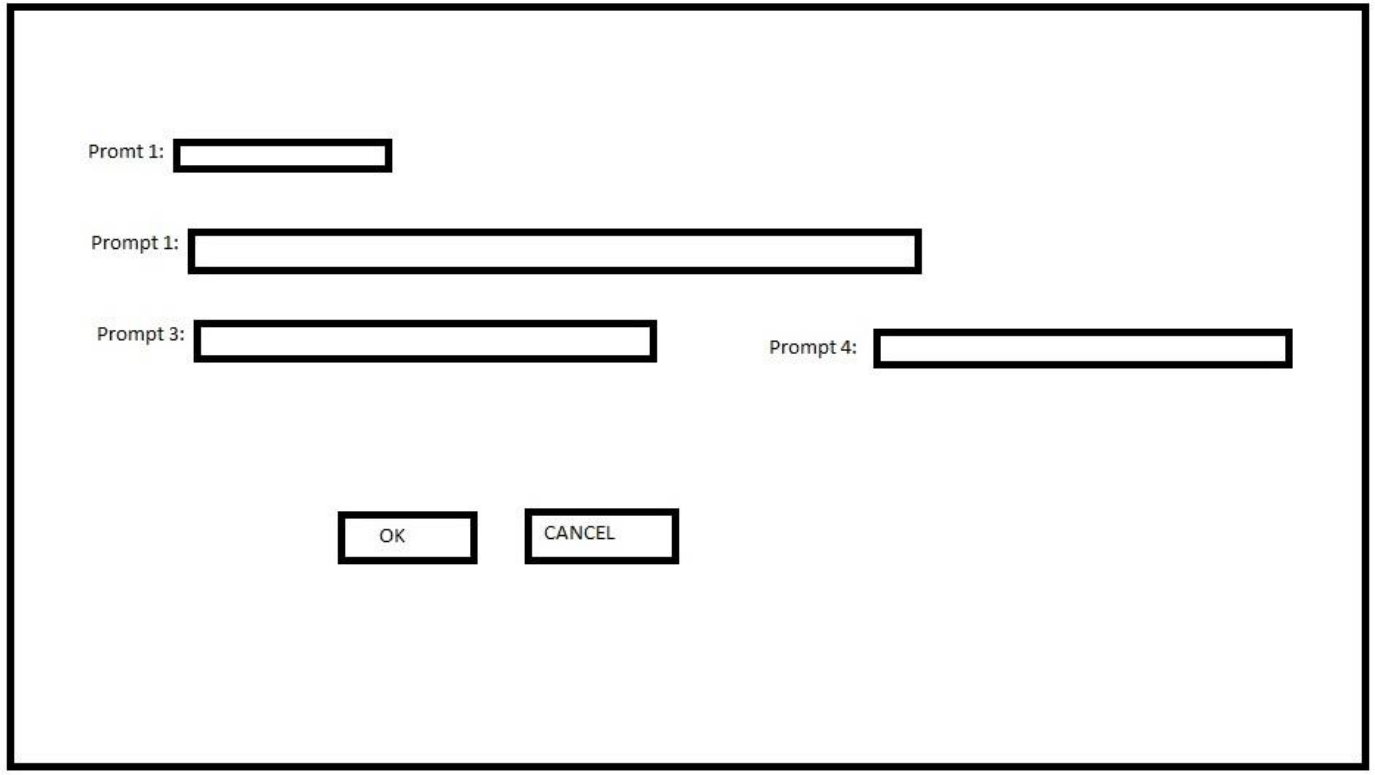
Rollback

ID	Name	Age	Salary
1	Milan	25	3000
2	Goran	23	1450
3	Marko	25	2000
4	Petar	27	5000
5	Jovan	29	3150

ID	Name	Age	Salary
1	Milan	25	3000
2	Goran	23	1450
3	Marko	25	2000
4	Petar	27	5000
5	Jovan	29	3150

```
> DELETE FROM CUSTOMERS WHERE AGE = 25;
> ROLLBACK;
```

Definisanje transakcije



Kako se osmišljava transakcija?

ŠEMATSKI PRIKAZ PROGRAMIRANJA JEDNE TRANSAKCIJE NAD BAZOM PODATAKA

```
Begin Transaction
    .....
    .....
    Insert 1 .....
    If ErrorStatus <> 0
        ROLLBACK
    ......
    ......
    Insert 2
    If ErrorStatus <> 0
        ROLLBACK
    .....
    .....
End Transaction
```

Savepoint

- **SAVEPOINT** – je tačka u transakciji, na koju možemo da se vratimo bez potrebe za rollback-om cele transakcije.
- Sintaksa za savepoint:

```
SAVEPOINT SAVEPOINT_NAME;
```

- Ova komanda služi samo za pravljenje ,,sigurne tačke“. Rollback-om se mogu poništiti ažuriranja transakcije do neke svapoint tačke.
- Sintaksa za rollback savepoint:

```
ROLLBACK TO SAVEPOINT_NAME;
```

Savepoint

ID	Name	Age	Salary
1	Milan	25	3000
2	Goran	23	1450
3	Marko	25	2000
4	Petar	27	5000
5	Jovan	29	3150

- SAVEPOINT SP1;

Savepoint created.

- DELETE FROM CUSTOMERS WHERE ID = 1;

1 row deleted.

- SAVEPOINT SP2;

Savepoint created.

- DELETE FROM CUSTOMERS WHERE ID = 2;

1 row deleted.

- SAVEPOINT SP3;

Savepoint created.

- DELETE FROM CUSTOMERS WHERE ID = 3;

1 row deleted.

Savepoint

- ROLLBACK TO SP2;

Rollback complete.

ID	Name	Age	Salary
2	Goran	23	1450
3	Marko	25	2000
4	Petar	27	5000
5	Jovan	29	3150

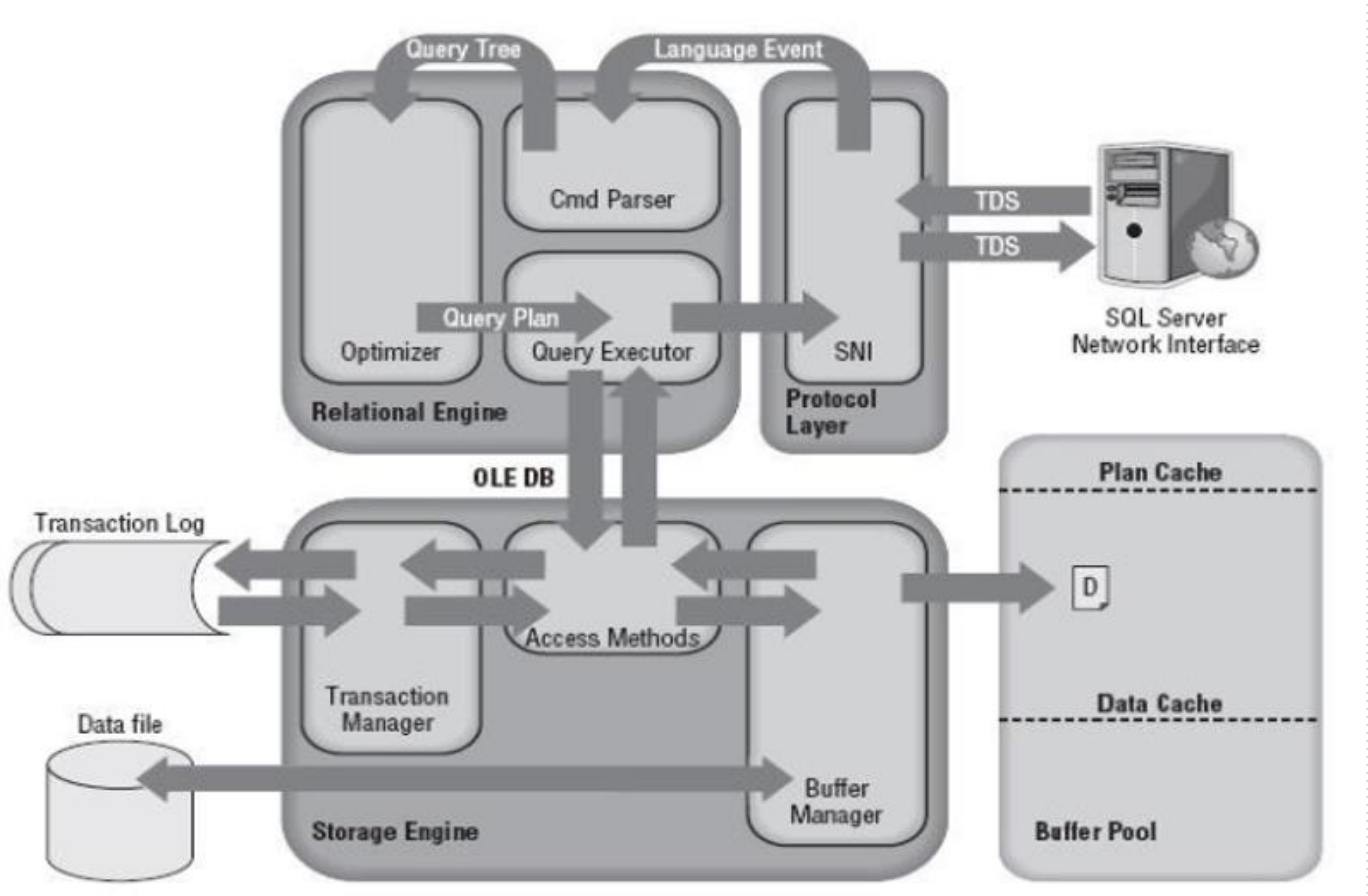
Release savepoint

- **RELEASE SAVEPOINT** – koristimo da obrišemo savepoint koji smo prethodno napravili.
- Sintaksa za release savepoint:
- RELEASE SAVEPOINT SAVEPOINT_NAME;
- Onog trenutka kad obrišete savepoint, više ne možete da koristite rollback nad transakcijama izvršenim od tog savepoint-a.

Set transaction

- **SET TRANSACTION** – komanda kojom započinjemo transakciju nad bazom podataka.
- Ovom komandom spefikujemo osobine određene transakcije.
- Na primer, možemo da kažemo da je transakcija read-only, ili da je transakcija read-write
- Sintaksa za set transaction:
- SET TRANSACTION [READ ONLY | READ WRITE];

Mesto Administratora transakcija u okviru SUBP-a



Žurnalizacija transakcija

- Žurnalizacija transakcija nad bazom podataka i "transaction logging" je proces beleženja svih promena koje se događaju u bazi podataka kako bi se omogućio oporavak podataka u slučaju neočekivanog prekida rada sistema (SUBP-a, OS ...) ili kvara hardvera, na nivou transakcije i na nivou baze podataka
- Koja je razlika između LOG-a transakcija i Žurnal fajla?
  - LOG transakcija i žurnala fajl su dva različita koncepta u kontekstu žurnalizacije transakcija u bazi podataka.
  - Log transakcija sadrži podatke svake transakcije koja se izvršava nad bazom podatakada bi se omogućilo vraćanje baze podataka u prethodno stanje u slučaju neočekivanih problema.
  - Žurnala fajl se koristi za smeštanje svih podataka o uspešno izvršenim transakcijama nad bazom podataka od trenutka uzete poslednje rezervne kopije baze podataka. Služi za oporavak (Restore) baze podataka.

Žurnalizacija transakcija (LOG fajl)

- ***Before image*** i ***After image*** su dva ključna koncepta u žurnalizaciji transakcija baze podataka.
- Ove koncepcije se često koriste u tehnologijama za zaštitu podataka u slučaju prekida rada i oporavka podataka.
- ***Before image*** žurnalizacija beleži stranice baze podataka pre ažuriranja, tako da se u slučaju greške u radu SUBP-a, operativnog sistema ili hardvera mogu oporaviti I obnoviti izmenjene stranice baze podataka iz žurnala.
- ***After image*** žurnalizacij, obično se koristi u relacionim SUBP. Beleži promene u žurnal pre nego što se te promene upišu u stvarnu bazu podataka.

Žurnalizacija ažuriranja Baze podataka

Vrste žurnalizacije

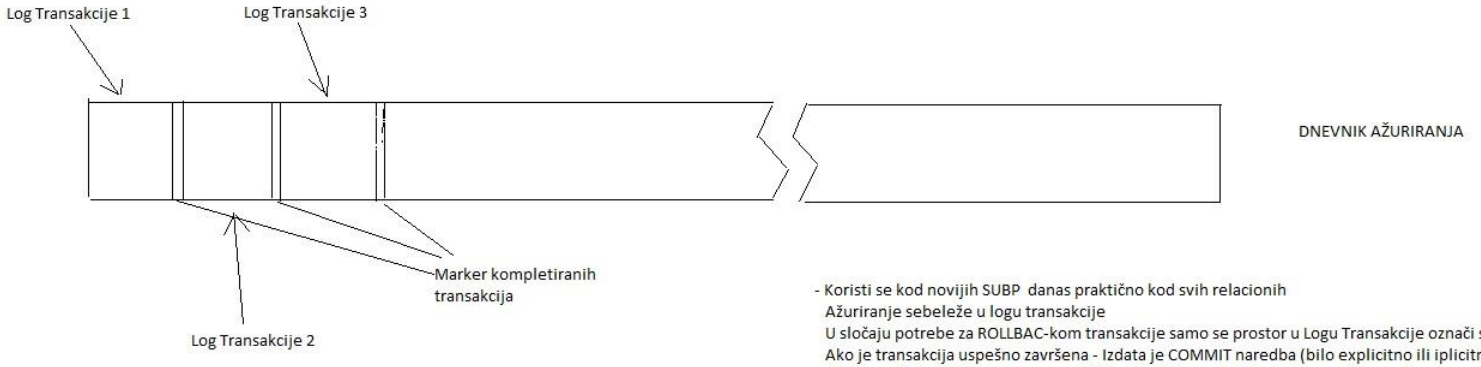
1. Before Image Žurnalizacija

2. After Image žurnalizacija

BEFORE IMAGE ŽURNALIZACIJA

- Korišćena kod starijih SUBP
  Kod SUBP zasnovanih na mrežnom modelupodataka i pre toga
  kod SUBP zasnovanih na hijerarhijskom modelu podataka
  - U log transakcija se upisuju Stranice BP pre ažuriranja, a
  - samo ažuriranje se vršilo u samoj BP
  - Ukoliko dođe do potrebe za ROLLBACK-om stranice pre ažuriranja BP
  su se vraćale u BP i na taj način se poništavalo ažuriranje
  - Posebno se vodio DNEVNIK AŽURIRANJA u drugoj datoteci u odnosu na log transakcije

AFTER IMAGE ŽURNALIZACIJA



- Koristi se kod novijih SUBP danas praktično kod svih relacionih
  Ažuriranje sebeleže u logu transakcije
  U sločaju potrebe za ROLLBAC-kom transakcije samo se prostor u Logu Transakcije označi slobodnim
  Ako je transakcija uspešno završena - Izdata je COMMIT naredba (bilo explicitno ili iplicitno
  - Vrši se ažiriranje BP na osnovu sadržaja LOGA Transakcije koja je uspešno završena
  - Logovi uspešno izvršenih transakcija se povezuju i praktično formiraju DNEVNIK Ažuriranja
  - Nema posebnog fajla za DNEVNIK Ažuriranja
  - Prilikom uzimanja rezervne kopija BP (BACKUP-a) prostor LOG-a Transakcija je praktično logički prazan
  Iako fizički prostor na disku ostaje zauzet, pa Fajl sistem može pokazivati da je fajl LOG-a Transakcija jako velik

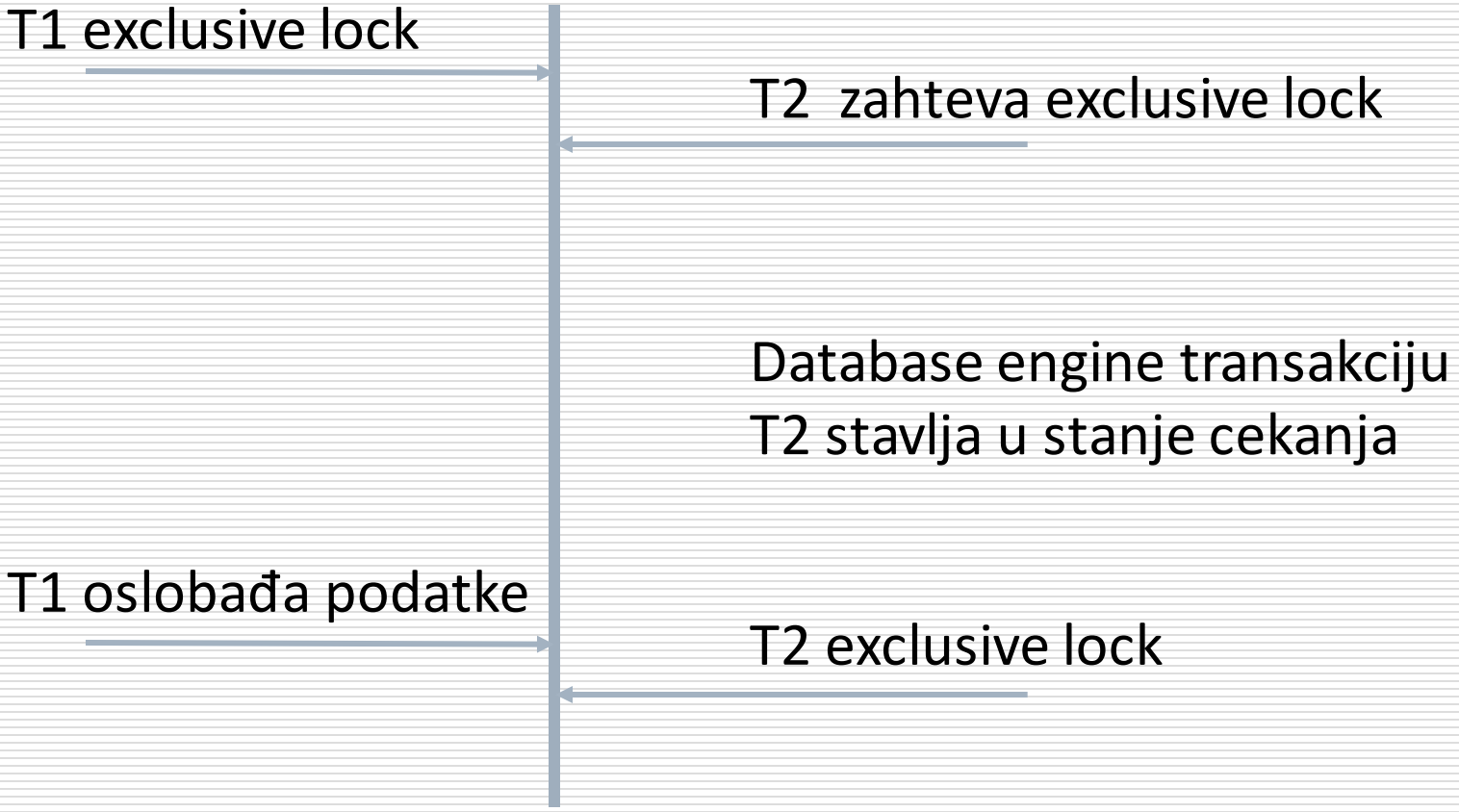
Zaključavanje baze podataka (locking in)

- **Zaključavanje** je mehanizam koji se koristi od strane SUBP-a da sinhronizuje pristup više korisnika istim podacima u isto vreme.
- Pre nego što transakcija počne da menja stanje podataka, prilikom čitanja ili ažuriranja, mora da se zaštiti od efekata druge transakcije koja pokušava da menja podatke istovremeno.
- Transakcija ovo realizuje slanjem zahteva za zaključavanjem tog dela podataka.
- Zaključavanje ima različite režime, kao što su **shared** (deljen) ili **exclusive** (isključivo za sebe).
- Zaključavanje definiše nivo zavisnosti koji transakcija ima nad određenim podacima.

Zaključavanje baze podataka

- Transakciji se ne može dodeliti pravo zaključavanja koje bi prouzrokovalo konflikt sa modom zaključavanja koji je druga transakcija već dobila.
- Ako transakcija zahteva mod zaključavanja koji će prouzrokovati konflikt sa zaključavanjem koje je već dodeljeno nad istim podacima, SUBP će zahtevanu transakciju staviti u status cekanja sve dok se ne oslobode podaci.
- Primer: T1 ima **exclusive** pravo nad podacima. T2 traži **exclusive** pravo nad istim podacima. Database engine transakciju T2 stavlja u stanje cekanja. T1 oslobađa podatke. Database engine dodeljuje **exclusive** pravo nad podacima za transakciju T2.

Zaključavanje baze podataka



Zaključavanje baze podataka

- Kada transakcija ažurira podatke, ona ih drži zaključanim štiteći ih od promena sve dok se transakcija ne zavrsi.
- Vreme koliko transakcija drži podatke zaključanim zavisi od podešavanja kojima je definisan nivo izolacije transakcije.
- Svako zaključavanje se oslobađa prilikom završetka transakcije (bilo u slučaju commit-a ili rollback-a).

Zaključavanje baze podataka

- Aplikacije obično ne zahtevaju zaključavnje podataka. Zaključavanjima se upravlja interno, komponentom Database Engine koji se zove **lock manager**.
- Kada istanca Database Engine procesira SQL upit, Database Engine Query procesor određuje kojim reursima može da se pristupa.

Zaključavanje baze podataka

- Query procesor odlučuje koji tipovi zaključavanja su neophodni da bi se zaštitio svaki resurs.
- Odlučivanje o tipu zaključavanja je zasnovano na tipu pristupa i nivou izolacije.
- Query procesor dalje zahteva određeno zaključavanje od ***lock manager-***a.
- Lock manager odobrava zaključavanje, samo ako ne postoji mogućnost da dodje do konflikta zbog neke druge transakcije.

Nivo izolacije transakcije

- ***Isolation levels*** su nivoi izolacije koji se koriste kako bi se regulisalo ponašanje transakcija u odnosu na ostale transakcije koje se izvršavaju u isto vreme.
- Omogućavaju definisanje nivoa vidljivosti podataka između različitih transakcija i utiču na to kako će se podaci prikazati kada se izvrši upit, u zavisnosti od toga koje su druge transakcije bile aktivne u isto vreme.
- Postoje četiri standardna nivoa izolacije transakcija, koji su definisani u ANSI/ISO SQL standardu. Oni su:
    1. Read Uncommitted (nivo izolacije čitanja nekomitovanih podataka)
    2. Read Committed (nivo izolacije čitanja komitovanih podataka)
    3. Repeatable Read (nivo izolacije ponovljivog čitanja)
    4. Serializable (nivo izolacije serijalizabilnosti)

Nivoi izolacije transakcije

1. Read Uncommitted (čitanje nekomitovanih podataka)
   - Najmanje restriktivan nivo izolacije.
   - Transakcija može da čita i prikaže podatke koji nisu još uvek komitovani u bazi podataka.
   - Može da dovede do prikaza nekonzistentnih podataka, jer se druga transakcija može izvršiti između dva čitanja istih podataka.
2. Read Committed (čitanje komitovanih podataka)
   - Najčešće koristi u bazama podataka.
   - transakcija može da čita samo one podatke koji su već komitovani u bazi podataka.
   - Obezbeđuje da će druga transakcija videti samo komitovane podatke.

Nivoi izolacije transakcije

3. Repeatable Read (ponovljivo čitanje)

- Transakcija može da čita podatke koji su bili prisutni u bazi podataka na početku transakcije.
- Bilo kakve promene koje se dešavaju u drugim transakcijama biti ignorisane,
- Transakcija ponašati kao da se ništa nije promenilo.

4. Serializable (serijalizabilnost)

- Najrestriktivniji nivo izolacije.
- Transakcije se izvršavaju jedna po jedna.
- Sve druge transakcije će biti blokirane dok se jedna transakcija ne završi. Ovo obezbeđuje najviši stepen konzistentnosti podataka
- Skup u smislu performansi

Repeatable Read (nivo izolacije ponovljivog čitanja)

- Obezbeđuje da se transakcija izvršava u okviru fiksne slike baze podataka, koja neće biti promenjena sve dok se transakcija ne završi.
- Druga transakcija neće biti u mogućnosti da izmeni podatke koji su uključeni u transakciju koja je u toku, sve dok ona ne završi.
- Ako je potrebno izvršiti nekoliko upita kako bi se izračunali rezultati transakcije, sve te upite treba izvršiti u okviru iste transakcije kako bi se osigurala konzistentnost podataka.
- Podaci se mogu čitati više puta tokom izvršavanja transakcije, ali ukoliko se upit ponovi, vrednosti koje se vraćaju će biti iste.
- Ovim nivoom izolacije, podaci se čitaju tako da se blokiraju ostale transakcije koje pokušavaju da izvrše ažuriranje tih istih podataka. na taj način se osigurava da se podaci neće promeniti tokom izvršavanja transakcije.

Granularnost i hijerarhija zaključavanja

- Database Engine ima multigranularno zaključavanje koje omogućava različitim tipovima resursa da budu zaključani određenom transakcijom.
- Da bi smanjili troškove zaključavnaja, Database Engine zaključava resurse automatski na nivou koji je prikladan tom zadatku.
- Zaključavajući manju granularnost resursa, kao što su redovi u tabeli, povećava se konkurentnost ali se troškovi povećavaju takođe jer više zaključavanja moramo obraditi istovremeno.
- Zaključavanje većih granularnosti, kao što su tabele, smanjićemo opšte troškove ali ćemo zaključati celu bazu, odnostno niko neće moći da pristupi bilo kojem delu te baze, čime ćemo smanjiti konkurentnost.

Granularnost i hijerarhija zaključavanja

- Primenljivo na: SQL Server.
- Database Engine mora da dobije multi level granularnosti da bi potpuno zaštitili podatke.
- Ove grupe zaključavanja na multi levelu granularnosti se zovu **hijerarhija zaključavanja**.
- Primer: Da bi potpuno zaštitili čitanje indeksa, Database Engine zahteva deljeno(shared) zaključavanje redova i (intent shared) deljeno zaključavanje stranica i tabela u bazi podataka.

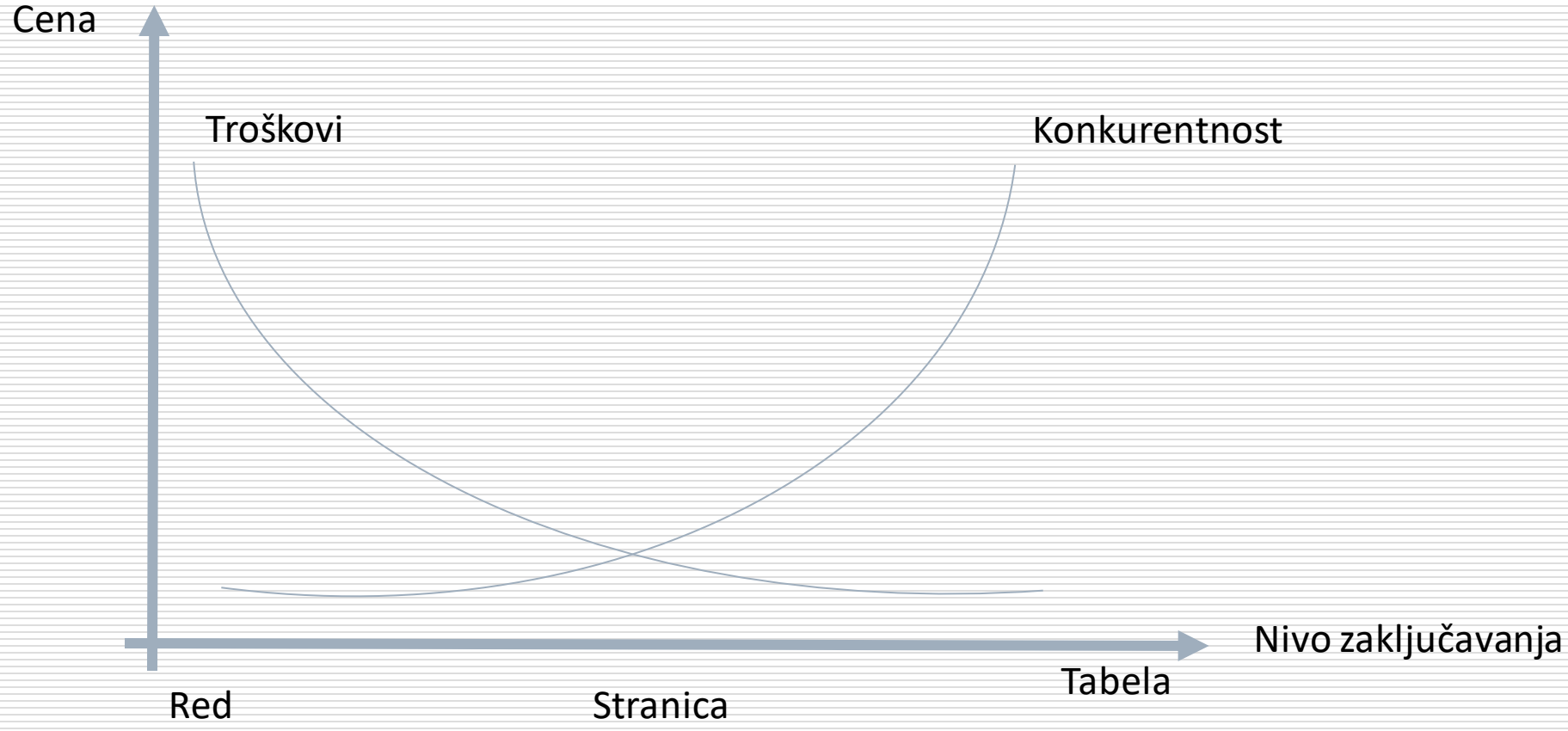
Granularnost i hijerarhija zaključavanja

- Nivoi zaključavanja:
- RID (Row Identifier) – zaključavanje jednog reda.
- KEY – zaključavanje reda unutar indeksa da bi zaštitili ključeve prilikom serijabilnih transakcija.
- PAGE – 8KB stranica u bazi podataka.
- EXTENT – grupa od 8 susednih stranica.
- HoBT – a heap or B-tree. Zaključavanje štiti B-tree ili heap stranice u tabeli koje još uvek nemaju grupisane indekse.

Granularnost i hijerarhija zaključavanja

- TABLE – zaključavanje cele tabele, uključujući podatke i indekse.
- FILE – fajl baze podataka.
- APPLICATION – zaključavanje na nivou aplikacije.
- METADATA – zaključavanje meta podataka.
- ALLOCATION_UNIT – jedinica alokacije (veličina klastera)
- DATABASE – zaključavanje cele baze podataka.

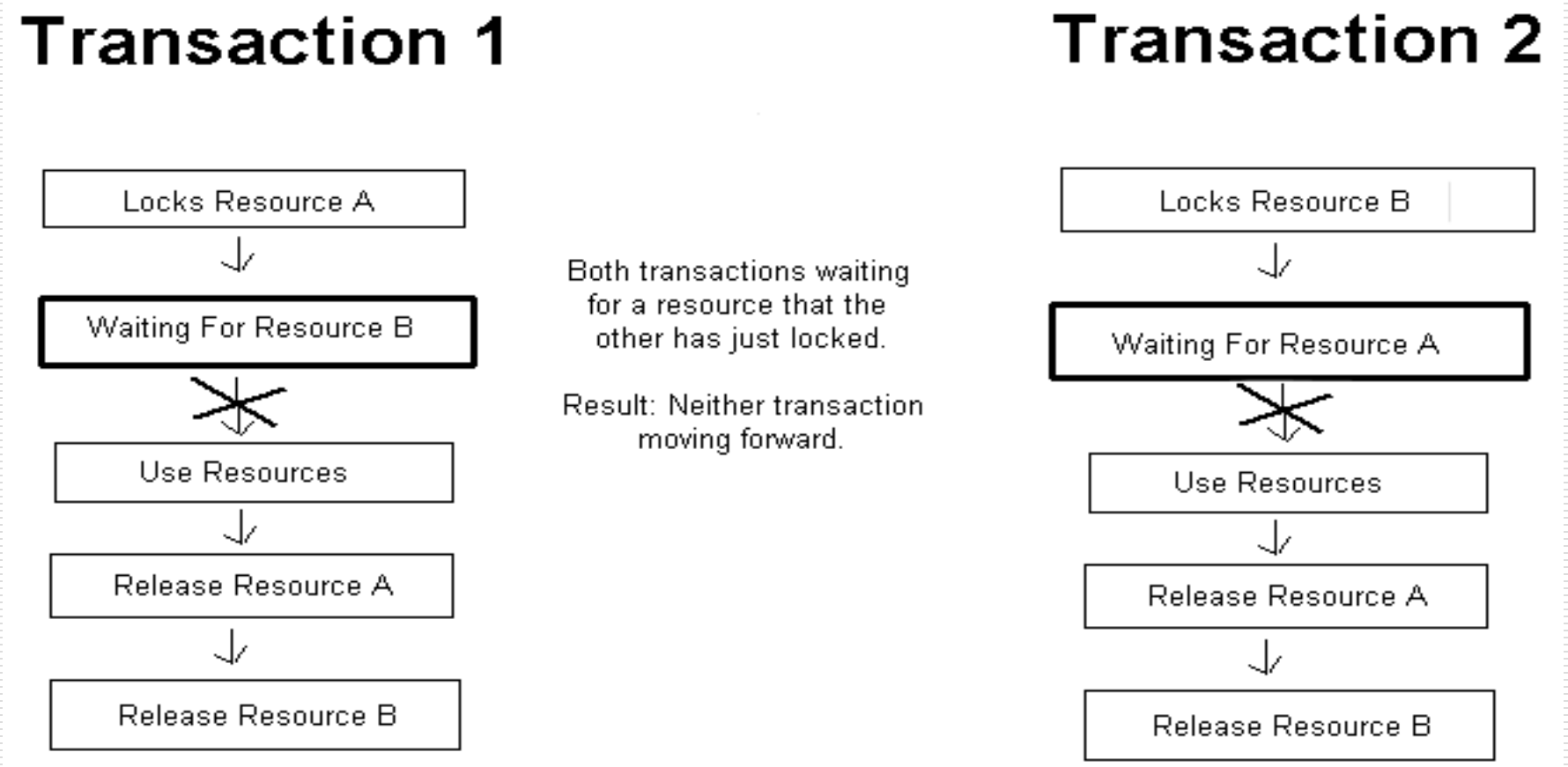
Granularnost i hijerarhija zaključavanja



Modovi zaključavanja

- Microsoft SQL Server Database Engine zaključava resurse koristeći različite modove za zaključavanje kojima odlučuje na koji način se može pristupati podacima od strane konkurentnih transakcija.
- Modovi zaključavanja koje Database Engine koristi:
- **Shared** (S) – koristi se za operacije čitanja koje ne menjaju podatke, kao sto je SELECT upit.
- **Update** (U) – koristi se za resurse koje možemo promeniti. Štiti od uobičajene forme deadlock-a koji se dešava kada više sesija čita, zaključava i potencijalno menja podatke istovremeno.

Modovi zaključavanja - deadlock



Modovi zaključavanja

- **Exclusive** (X) – koristi se za operacije koje ažuriraju podatke kao što su INSERT, UPDATE ili DELETE. Obezbeđuje da ne možemo vršiti višestruko ažuriranje istih podataka istovremeno.
- **Intent** – koristi se da uspostavi hijerarhiju zaključavanja. Tipovi intent zaključavanja su: intent shared (IS), intent exclusive (IX) i shared intent exclusive (SIX).
- **Schema** – koriste se kada se operacije zavisne od šeme tabele izvršavaju. Tipovi schema zaključavanja su: schema modification (Sch-M) i schema stability (Sch-S).

Modovi zaključavanja

- **Bulk Update** (BU) – koristi se prilikom kopiranja podataka u tabele i kada je TABLOCK hint naveden.
- **Key-Range** - štiti redove koji se čitaju od strane upita prilikom serijabilnih transakcija sa određenim nivoom izolacije. Obezbeđuje da druge transakcije ne mogu da dodaju redove koji će biti kvalifikovani za upit serijabilnih transakcija ako su upiti pokrenuti ponovo.

Modovi zaključavanja - kompatibilnost

Mode	NL	IS	IX	S	SIX	X
NL	Yes	Yes	Yes	Yes	Yes	Yes
IS	Yes	Yes	Yes	Yes	Yes	No
IX	Yes	Yes	Yes	No	No	No
S	Yes	Yes	No	Yes	No	No
SIX	Yes	Yes	No	No	No	No
X	Yes	No	No	No	No	No