

# SQL

---

## Transakcije

# Pojam transakcije

---

- Transakcije su vrlo bitne u svetu baza podataka
- Transakcija u realnom svetu
  - Ukoliko prodavcu knjiga platimo, očekujemo da dobijemo knjigu.
  - Transakcija je ukoliko su se obe stvari desile
  - Ukoliko smo dali novac očekujemo knjigu, ukoliko prodam knjigu očekujem novac za nju
- U svetu računara klasičan primer transakcije je bankarski sistem.
  - Ulogujemo se na naš račun preko Weba i želimo da prebacimo novac sa jednog računa na drugi.
  - Ova akcija zahteva dve promene nad podacima, sa jednog računa oduzimamo novac a na drugom računu dodajemo novac.
  - Transakcija je uspešna ukoliko su se obe stvari izvršene, ukoliko jedna stvar ne uspe, sistem vraća podatke na početno stanje

# Osobine transakcije

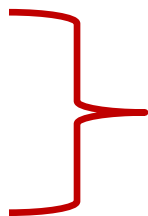
---

- Termin koji se vrlo često koristi u radu sa transakcijama je ACID i ugrađen je DBMS
  - Atomic
    - Zahteva da se sve akcije u transakciji izvrše ili sistem vraća na originalno stanje.
    - Razlog zašto se sve akcije u transakciji ne izvrše je nestanak električne energije ili nedovoljno prostora ili ...
    - Atomic znači sve ili ništa
  - Consistent
    - Transakcijom baza podataka iz jednog validnog stanja prelazi u drugo na osnovu pravila u bazi
  - Isolated
    - Odnosi se na to da podatak koji je uključen u transakciji bude zaključan za vreme trajanja transakcije, tj. nesme da se dozvoli drugom sistemu da menja isti podatak
  - Durable
    - Garantuje izvršenje transakcije čak i ukoliko se desi neki otkaz

# TRANSAKCIJE – Isolated Osobina

---

ID	Nickname	Balance	...
1	Joint	\$9000	...
2	Alice	\$1050	
3	Bob	\$45	
...	...	...	



- Tri računa
  - Joint - zajednički račun
  - Alis i Bob - odvojeni računi
- Posmatraju se koraci u transferu novca između Joint računa i Alis računa

Prikaži saldo **Joint** računa (\$10000)

Prikaži saldo **Alis** računa (\$50)

Promeni saldo na **Joint** računu (\$10000) - \$1000

Promeni saldo na **Alis** računu (\$50) + \$1000

# TRANSAKCIJE – Isolated Osobina

Javiće se konflikt jer dva programa istovremeno rade nad istim podatkom, na Joint računu treba da bude \$8000

ID	Nickname	Balance	
1	Joint	\$9000	...
2	Alice	\$1050	
3	Bob	\$1045	
...	...	...	

Slučaj kada Alis i Bob istovremeno rade operacije sa svog i Joint računa.

Bob veruje da na Joint racunu i dalje ima \$1000 jer je u prvoj iteraciji prilikom prikazivanja dobijo taj podatak

## Alis

Prikaži saldo **Joint** računa (\$10000)

Prikaži saldo **Alis** računa (\$50)

Promeni saldo na **Joint** računu (\$10000) - \$1000

Promeni saldo na **Alis** računu (\$50) + \$1000

## Bob

Prikaži saldo **Joint** računa (\$10000)

Prikaži saldo **Bob** računa (\$45)

Promeni saldo na **Joint** računu (\$10000) - \$1000

Promeni saldo na **Bob** računu (\$45) + \$1000

# Transakcija – rešenje problema

- Potrebno je nekoliko koraka učiniti jedinstvenim tj grupisati u jedan blok koji je nedeljiv praveći transakciju.
- Ukoliko se u toku transakcije javi problem(nestanak el. energije) sistem se vraća na stanje pre transakcije.
- Transakcije i dalje ne rešavaju *race condition* problem tj. transakcije koje se izvršavaju u isto vreme.
  - Slučaj kada je Bob u fazi čitanja podataka a Alis u fazi izmene podataka (*dirty read*)

## Alis

### BEGIN TRANSACTION

Prikaži saldo Joint računa (\$10000)

Prikaži saldo Alis računa (\$50)

Promeni saldo na Joint računu (\$10000)- \$1000

Promeni saldo na Alis računu (\$50) + \$1000

### COMMIT

## Bob

### BEGIN TRANSACTION

Prikaži saldo Joint računa (\$10000)

Prikaži saldo Bob računa (\$45)

Promeni saldo na Joint računu (\$10000)- \$1000

Promeni saldo na Bob računu (\$45) + \$1000

### COMMIT

# Transakcija – Pesimistično zaključavanje

- Transakcija nije dovoljna za rešenje problema dirty read, već je potrebno obezbediti i *zaključavanje* od istovremene promene istog podatka.
- Ideja je da čim transakcija startuje podatak sa kojim transakcija radi se zaključava sve dok se ne završi transakcija (commit).

ID	Nickname	Balance	...
1	Joint	\$8000	...
2	Alice	\$1050	
3	Bob	\$1045	
...	...	...	

Automatski se zaključava Joint račun

**Alis**

## **BEGIN TRANSACTION**

Prikaži saldo Joint računa (\$10000)

Prikaži saldo Alis računa (\$50)

Promeni saldo na Joint račun ( \$10000)- \$1000

Promeni saldo na Alis račun ( \$50) + \$1000

**COMMIT**

Bob čeka dok se Joint račun(čelija) ne otključa, transakcija sa leve strane ne završi koja traje 10 deo sekunde

**Bob**

## **BEGIN TRANSACTION**

Prikaži saldo Joint računa (\$10000)

Prikaži saldo Bob računa (\$45)

Promeni saldo na Joint račun ( \$10000)- \$1000

Promeni saldo na Bob račun ( \$45) + \$1000

**COMMIT**

# Transakcija – Optimistično zaključavanje

- Pesimistično zaključavanje u našem slučaju je dobro za Alis ali ne i za Boba koji je morao da čeka
- Ovakav način zaključavanja može dovesti do toga da veliki broj korisnika treba da čeka ili do pojave greške.
- Međutim sama transakcija ne mora da znači da će doći do menjanja podataka već samo do čitanja.
- Optimistično zaključavanje dozvoljava da se više transakcija izvršavaju istovremeno sa ciljem da neće doći do konflikta

**Alis**

**BEGIN TRANSACTION**

Prikaži saldo Joint računa (\$10000)

Prikaži saldo Alis računa (\$50)

Promeni saldo na Joint račun ( \$10000)- \$1000

Promeni saldo na Alis račun ( \$50) + \$1000

**COMMIT**

DBMS je otkrio konflikt sa drugom transakcijom, vraća se na početak transakcije

**Bob**

**BEGIN TRANSACTION**

Prikaži saldo Joint računa (\$10000)

Prikaži saldo Bob računa (\$45)

Promeni saldo na Joint račun ( \$10000)- \$1000

Greška – dirty read je otkrivena - Rollback