# **SBP**

## Realizacija ograničenja šeme RBP putem SUBP

Predavanje 0.

### (Uvod) Realizacija integritentne komponente šeme BP

* + **Cilj** je da sva naša ograničenja organizujemo na nivou servera BP putem mehanizma SUBP, ideja je da prakticno zastitimo sve naše podatke, tako da odakle god da im pristupimo da ne možemo da ih narušimo.
  + **Kontrola ograničenja, implementiranih na nivou SUBP je centralna** to u suštini znači da ni jedan softver ni program ne može spolja da ih naruši, korisnici nisu svesni ograničenja sve dok ih ne naruše. Primer prolazimo kroz aerodrom i mi bivamo iskontrolisani a da nismo ni svesni dok ne narušimo neko ograničenje, isto funkcioniše i SUBP. U slučaju da narušimo nešto SUBP će obaviti jednu od ove dve operacije:
    - obavestiti nas o greski (korisnicki program i korisnika nas) i prekinuti izvršavanje
    - da nekim aktivnim mehanizmom obavi dalje operacije nad BP da bi tu BP vratio u konzistentno stanje
  + **Implementacija ograničenja BP (na serveru BP),** moramo da definiše parametre u vezi ograničenja (tj da ih saopštimo serveru), parametri su:
    - definiše se ograničenje datog tipa
    - definišu se operacije tj. događaji koji mogu dovesti do narušavanja ograničenja
    - za svaku operaciju je potebno definisati akciju koju ćemo preduzeti da bi očuvali konzistentnost u slučaju narušavanja ograničenja
    - svako ograničenje je vezano za određenja obeležja i određene šeme relacija šeme BP
    - operacije(događaji) koji mogu dovesti do narušavanja ograničenja:
      * Upis (nove torke u relaciju)
      * Brisanje (neke postojece torke)
      * Modifikacija (postojece torke u relaciji)

Pitanje: Zašto upiti nisu kritična operacija? Upiti ne menjaju stanje, samo konstatuju koji su to podaci u BP, ako su upiti ispravni sve ce biti ok ako nisu ispravni samo će pokazati da nisu ispravni.

Pitanje: Kada možemo da donesemo pretpostavku da će nam upiti vratiti uvek ispravne podatke ? Ako krenemo od početka i ako kontrolišemo svaki upis, brisanje i modifikaciju, posledično svaki naš upit će vratiti konzistentno ispravne podatke. U životu ovo baš nije uvek tako pošto često preuzimamo nasleđene sisteme.

* + - Akcije očuvanja konzistentnosti BP u slučaju pokušaja narušavanja ograničenja, vezuju se sa svaku operaciju koja može narušiti ograničenje, dele se na:
      * Aktivne, obezbeđuju automastsko sprovođenje daljih operacija ažuriranja nad BP, koje će obezbediti očuvanje konzistentnosi BP
      * Pasivne, sprečavaju operaciju koja bi narušila ograničenje
      * Kombinovane, pod određenim uslovima pasivne pod određenim uslovima aktivne
    - Ograničenja se implementiraju pomoću mehanizama SUBP, imaćemo način da definišemo ograničenje i moraćemo da saopštimo DBMsu kako smo definisali ograničenje, imamo kritične operacije koje mogu narušiti ograničenje opet moramo saopštiti DBMSu koje su kritične operacije, i za savaku kritičnu operaciju moramo saopštiti koja akcija očuvanja konzistentnosti BP će se primeniti
  + **Kontrola ograničenja šeme BP,** SUBP pokreće mehanizme kontrole važenja ograničenja
    - automatski
    - nakon izvođenja kritične operacije za ograničenje
  + **Prednosti**:
    - Automatksa kontrola implementiranih ograničenja na nivou SUBP
    - Imamo obezbeđenu konzistentnost BP u svakom trenutku
    - Standardizacija načina za implementaciju ograničenja (sql2, sql3, etc...)
    - Kada postavimo implementaciju ograničenja na nivo servera baze podataka onda možemo da kažemo da nemamo dodatnih razloga zašto bi realizovali ograničenja unutar samih korisničkih programa na srednjem sloju, ili na klijentskom sloju. **Znači implementiramo ograničenja jednom, a ne milion puta na različitim aplikacijama.**

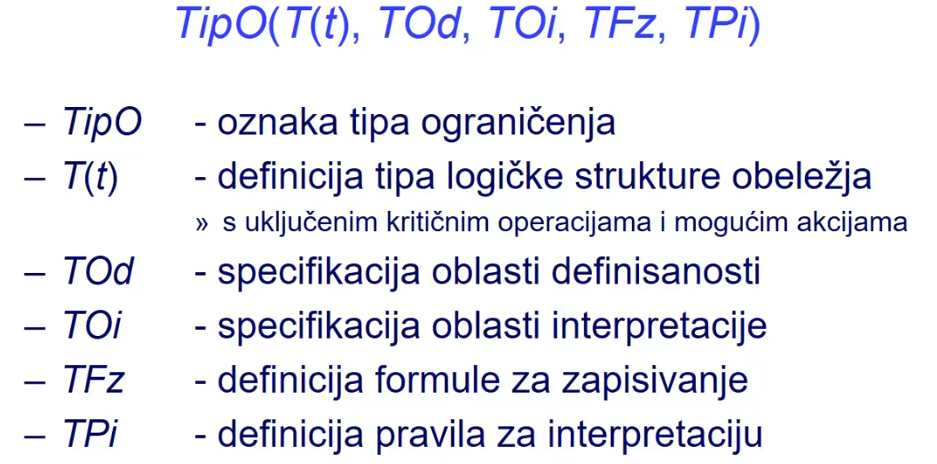
Pitanje: Zašto je ovo jako dobro? Rasterećujemo programere i olakšavamo samu izradu aplikacije kao i njeno održavanje koje jako puno košta.

* + **Nedostaci:** 
    - Povišen stepen zavisnosti šeme BP od proizvođača, tipa i verzije SUBP-a, ne podržavaju svi sistemi za upravljanje BP standarde na isti način, standardi generalno nisu idealni, to znači mi ako želimo da koristimo kapacitete DBMS-a za implementaciju ograničenja na nivou servera BP nužno ćemo istrčavati izvan onoga što definiše standard, znači ako želimo da migriramo bazu sa jednog na drugi DBMs moraćemo da preradimo te delove programskog koda, postajemo više zavisni od šeme BP i proizvodjaca tipa i verzije. **Zato je izbor DBMS-a strateski potez jedne kompanije!**
    - Ostaje uvek potreba za realizacijom nekih ograničenja unutar programa, zbog obezbedjenja pogodnosti programa za upotrebu, i dvostruke implementacije i kontrole ograničenja. Recimo ako pravimo interfejse bolje pogodnosti (HCI priča) obično želimo da obavestimo korisnika da pravi grešku u trenutku kada je pravi a ne kada klikne submit. Onda imamo potrebu da ista ograničenja implementiramo i na frontu a ne samo na serveru BP.

Predavanje 1.

## Specifikacija tipa ograničenja

* + **Karakteristike tipa ograničenja u modelu podataka (MP)**
    - Oblast definisanosti, to je tip logičke strukture nad kojom se ograničenje definiše
    - Oblast interpretacije, to je tip logičke strukture nad kojom se ograničenje interpretira
    - Formalizam za zapisivanje, to je nekakv parametrizovani šablon na osnovu kog pišemo ograničenja
    - Pravilo za interpretaciju odnosn ovalidaciju
    - Skup kritičnih operacija nad bazom podataka koje mogu dovesti do narušavanja ograničenja datog tipa, mi idemo sa pretpostavkom da imamo praznu bazu podataka
    - Skup mogućih akcija, definiše se za svaku kritičnu operaciju
  + **Specifikacija tipa ograničenja u MP**



* + - Sve komponente specifikacije TipO treba da budu specificirane formalno( u cilju njihove lakše programske implementacije). U ove potrebe potrebno je koristiti koncepte nekog konkretnog MP (mi ćemo koristiti primer relacionog modela podataka).
    - TipO – oznaka tipa ograničenja,
      * to je nekakav niz znakova, ne poste neka konkretna pravila kako zadajemo taj niz znakova ali očekujemo da taj niz znakova jedinstveno identifikuje svaki uvedeni tip ograničenja, formira se sasvim slobodno na osnovu naziva tipa ograničenja u datom MP

## Tipovi ograničenja u RMP

* + **Mogući tipovi ograničenja u RMP**

