REDUDANSA ŠEME RBP

Suština ovog vida restrukturiranja, koje se sprovodi nakon redukcije.šeme RBP, jeste u dodatnom smanjenju broja i složenosti operacija upita, i to:

- ✔ Smanjenje/uklanjanje operacija spajanja.
- ✔ Smanjenje/uklanjanje operacija svođenja.

Za razliku od redukcije, za redudansu je karakteristično uvođenje novih (redudantnih) podataka u šemu RBP koji se inače mogu dobiti odgovarajućim upitima nad prvobitnom šemom RBP. To uvođenje redudantnih podataka se može sprovesti na dva načina:

- ✔ Dodavanje novih kolona postojećim tabelama.
- ✔ Uvođenje novih tabela.

Cena koja se plaća pri uvođenju redudanse jeste usložnjavanje postupaka održavanja podataka, pošto uz osnovne postupke treba predvideti i postupke usklađivanja redudantnih podataka sa izvornim podacima iz kojih se izvode. To se može ostvariti na više načina:

- ✔ Dopunom aplikativne logike na nivou iznad servera baze podataka.
- ✔ Preko SQL procedure na nivou servera baze podataka.
- ✔ Preko SQL okidača.

Po osnovnoj nameni postoje dva vida redudanse:

- ✔ Replikativna redudansa.
- ✔ Svodna redudansa.

REPLIKATIVNA REDUDANSA

Kod replikativne redudansu je karakteristično sledeće:

- ✔ Dodatni podaci su izvorni - nisu izvedeni svođenjem nego su kopije (replike) izvornih podataka koji postoje u neredukovanoj bazi podataka;
- ✔ Izvorni dodatni podaci nisu dobijeni svođenjem u neredukovanoj bazi podataka.

Strukturna implementacija

Neka je red željenog upita strukture

$$D = < D1, D2, \ldots , DN >$$

gde su sa Di označeni izvorni podaci koji su po karakteru opisni (deskriptori).

U prvom koraku traži se tabela T u kojoj se javlja neki maksimalni deo Dx željenih podataka pri čemu je Dy deo željenih podataka koji se tu ne javlja

$$Dx = < Dx1, \ldots , DxP > \qquad Dy = < Dy1, \ldots DyQ >$$

Dalji postupak se svodi na sledeće

```
IF ( Dx jednako D )
   Redudansa: nije potrebna - T ima sve
ELSE
   IF ( jednom Dx odgovara jedno Dy )
      Redudansa: dopuna tabele T sa Dy
   ELSE
      Redudansa: nova tabela U(D)
   ENDIF
ENDIF
```

Manipulativna implementacija

Preko okidača baze podataka. Postoje dva slučaja:

- ✔ Za slučaj uvođenja nove tabele okidač treba da obezbedi unos reda u U ako on već ne postoji;
- ✔ Za slučaj dopune postojeće tabele T okidač treba da obezbedi ažuriranje redudantnih podataka.

U slučaju da unutar D postoje funkcijske zavisnosti Dr -> Ds, za svaku takvu zavisnost treba obezbediti okidač koji za slučaj izmene izvornog Ds ažurira replicirano Ds.

SVODNA REDUDANSA

Kod svodne redudanse su moguće dve situacije:

- ✔ Dodatni podaci su svodni - nisu kopije (replike) izvornih podataka koji postoje u neredukovanoj bazi podataka nego su izvedeni iz njih;
- ✔ Dodani podaci su izvorni ali su dobijeni svođenjem u neredukovanoj bazi podataka (ovo je slučaj kada redudansa iskazuje neki odnos veze ili sličnog).

Strukturna implementacija

Neka je red željenog upita strukture

$$D = < D1, D2, \ldots , DN > < K1,..,KM >$$

gde su sa Di označeni izvorni podaci koji su po karakteru opisni (deskriptori) a sa Kj kvantifikovani podaci koji se dobijaju svođenjem.

U prvom koraku traži se tabela T u kojoj se javlja neki maksimalni deo Dx željenih podataka pri čemu je Dy deo željenih podataka koji se tu ne javlja

$$Dx = < Dx1, \ldots , DxP > \qquad Dy = < Dy1, \ldots DyQ >$$

Dalji postupak se svodi na sledeće (sa K su označeni kvantifikovani podaci):

```
IF ( Dx jednako D )
   Redudansa: dopuna tabele T sa K
ELSE
   IF ( jednom Dx odgovara jedno Dy )
      Redudansa: dopuna tabele T sa Dy i K
   ELSE
      Redudansa: nova tabela U(D,K)
   ENDIF
ENDIF
```

Kod dopune tabele T za sve podatke u okviru K treba obezbediti odgovarajuće inicijalne vrednosti (DEFAULT klauzula u CREATE TABLE).

Manipulativna implementacija

Preko okidača baze podataka. Postoje dva slučaja:

- ✔ Za slučaj uvođenja nove tabele okidač treba da obezbedi unos reda <D,K> u U ako vrednost D ne postoji u U, pri čemu se za K uzimaju inicijalne vrednosti (1 ako je u pitanju brojanje, sabirak ako je u pitanju sabiranje, itd.); ako vrednost D postoji u U vrši se ažuriranje K u odgovarajućem redu;
- ✔ Za slučaj dopune postojeće tabele T okidač treba da obezbedi ažuriranje redudantnih podataka K za red sa odgovarajućom vrednošću D.

U slučaju da unutar D postoje funkcijske zavisnosti Dr -> Ds, za svaku takvu zavisnost treba obezbediti okidač koji za slučaj izmene izvornog Ds ažurira replicirano Ds.

ELEMENTI NOTACIJE

Strukturna implementacija

Za dopunu postojeće tabele T:

```
T + { Kolona = InicijalnaVrednost } ,..
```

Za uvođenje nove tabele U:

```
U ( Kolona ,.. )
```

Manipulativna implementacija

Specifikacija okidača se zadaje u šest delova:

```
Aktivacija:     BEFORE|AFTER

Dogadjaj:       INSERT|DELETE|{UPDATE[(Kolona,..)]} Tabela

Opseg:          ROW|STATEMENT

Referenca:      [OLD NadimakO][NEW NadimakN]

Uslov:          X|OpisUslova

Akcija:         SpecifikacijaAkcije
```

`SpecifikacijaAkcije` je iz tri dela odvojena podcrtavanjem:

```
SpecifikacijaReferenci
----------------------
SpecifikacijaOcitavanja
-----------------------
SpecifikacijaObrade
```

`SpecifikacijaReferenci` je u formi (X je O ili N):

```
[NadimakX.Kolona,..]
```

`SpecifikacijaOcitavanja` je u formi:

```
{NadimakX.Kolona|vVarijabla},.. > Tabela > vVarijabla
```

`SpecifikacijaObrade` može da sadrži uslovljavanje IF , IF-ELSE ili IF-ELSEIF-... . Uslov se navodi opisno. Unutar obrade mogu po potrebi da se jave dodatna očitavanja. Obrada nad nekom tabelom zadaje se u formi

```
Podatak ?|>
...
{INSERT Tabela}|{DELETE Tabela}|{UPDATE Tabela
                                 Izmena
                                 ...}
```

gde `Podatak` može biti `NadimakX.Kolona` ili `vVarijabla`, `?` označava selektor reda i `>` ulazni podatak. `Izmena` se zadaje kao

```
(Kolona = NovaVrednost)
```