UDC: 004.6 INFO M: str. 36-43

# GENERISANJE SQL OPISA ŠEME BAZE PODATAKA ZA RAZLIČITE SUBP
# GENERATING SQL SPECIFICATIONS OF A DATABASE SCHEMA FOR DIFFERENT DBMSS

Slavica Aleksić, Ivan Luković

REZIME: U radu je prikazan SQL generator, koji je deo IIS*Case-a – softverskog alata za automatsko projektovanje i generisanje šema baza podataka. SQL generator generiše implementacioni opis šeme baze podataka po ANSI SQL:2003 standardu, kao i za komercijalne sisteme za upravljanje bazama podataka (SUBP) MS SQL Server i Oracle Server. U radu će biti prezentovane neke od osobina SQL generatora koje se odnose na implementaciju šema relacija i ograničenja baze podataka za izabrane SUBP i njihove programske jezike T-SQL i PL/SQL. Na izabranom primeru, daje se poređenje osobina T-SQL-a i PL/SQL-a koje se odnose na implementaciju ograničenja baze podataka, kao i analiza podrške i odstupanja MS SQL Server i Oracle SQL jezika od standarda ANSI SQL:2003.
KLJUČNE REČI: Projektovanje i implementacija šeme baze podataka, ograničenja baze podataka, SQL generator, IIS*Case.

ABSTRACT: In the paper we present an SQL generator, which is integrated into IIS*Case -- a software tool for automated design and generating of database schemas. The SQL generator can generate implementation specifications of a database schema according to ANSI SQL:2003 standard, as well as for current database management systems (DBMSs) MS SQL Server and Oracle Server. We discuss some specific features of the SQL generator that concern implementation of relation schemes and database constraints for the selected DBMSs, under the server programming languages T-SQL and PL/SQL. Based on a selected example, we compare the features of T-SQL and PL/SQL in the course of implementation of database constraints, and also analyze the conformance of MS SQL Server and Oracle SQL to ANSI SQL:2003.
KEY WORDS: Database Schema Design and Implementation; Database Constraints; SQL Generator; IIS*Case.

## 1. UVOD

Integrated Information Systems*Case (IIS*Case), V.6.2 je alat koji pripada klasi softverskih proizvoda namenjenih za podršku ili automatizaciju što većeg broja aktivnosti razvoja drugih programskih proizvoda. Osnovni koncepti i formalizmi na kojima je zasnovan IIS*Case nastajali su od kraja osamdesetih godina na Fakultetu tehničkih nauka u Novom Sadu. Izrađene su brojne verzije IIS*Case-a, u različitim programskim okruženjima. Poslednja verzija programa obuhvata do sada najveći broj tih koncepata i algoritama. Detaljne informacije o IIS*Case-u mogu se pronaći u [5, 8, 13, 14].

SQL generator je deo ovog komplesnog softvera. Obezbeđuje automatsko generisanje SQL koda, koristeći specifikaciju nastalu projektovanjem šeme baze podataka. SQL generator obezbeđuje korisniku inteligentnu pomoć u radu, da i bez poznavanja sintakse SQL-a kao i mehanizama za implementciju ograničenja, dobije SQL kod za kreiranje tabela, indeksa, sekvenci, procedura, funkcija i okidača. U ovoj realizaciji, SQL generator obezbeđuje korisniku da bira generisanje implemetacionog opisa šeme baze podataka po sintaksi:

- standarda ANSI SQL:2003 [4],
- jezika Microsoft T-SQL MS SQL Servera 2000/2005 [6, 7] i
- jezika Oracle PL/SQL Oracle Servera 9i/10g [10, 11].

Do danas je u svetu razvijen svakako ne mali broj CASE alata, različitih mogućnosti i namene, u koje su integrisani generatori SQL opisa šeme baze podataka. Neki od njih su navedeni u literaturi [2, 12]. Ideja razvoja SQL generatora u alatu IIS*Case je bila ne samo u tome da se obezbedi funkcionalnost i mogućnosti koje imaju drugi alati iste namene, već i da se podrži realizacija nekih specijalnih ograničenja šeme baze podataka, kao i nekih posebnih opcija, koje autori nisu pronašli u drugim dostupnim alatima, a pokazuje se da mogu biti od koristi u praksi projektovanja i realizacije baza podataka. [15]

Cilj ovog rada je da se prikažu neke specifičnosti SQL generatora koje se odnose na implementaciju šema relacija i kontrolu integriteta baze podataka na izabranim sistemima za upravljanje bazama podataka (SUBP). Kroz izabrane primere implementacije ograničenja baze podataka biće date uporedne karakteristike jezika T-SQL i PL/SQL. Biće, takođe, analizirana podrška standarda ANSI SQL:2003 za izabrane SUBP, saglasno rezultatima, prikazanim u [1].

Pored uvoda i zaključka, rad ima pet delova. U drugom i trećem delu analizirani su usaglašenost i odstupanja od standarda za Oracle Server i MS SQL Server, u onom delu koji je bio važan za realizaciju SQL generatora. U četvrtom delu, ukratko su diskutovani praktični aspekti primene SQL generatora. U petom delu prikazane su specifičnosti generisanja SQL opisa šeme baza podataka na primeru implementacije jednog izabranog ograničenja, dok su u šestom delu date uporedne karakteristike ova dva SUBP-a.

## 2. MEHANIZMI ZA REALIZACIJU OGRANIČENJA ZA ORACLE SERVER 9I/10g

Kada je u pitanju implementacija strukture i ograničenja baze podataka, Oracle Server (verzija 9i i 10g), pored SQL-a, predviđa i upotrebu posebnog proceduralno orijentisanog jezika koji se zove Procedural Language/Structured Query Language (PL/SQL). U ovom delu rada, diskutuje se usaglašenost koncepata koje podržava Oracle Server sa standardom ANSI SQL:2003, u delu koji se odnosi na implementaciju logičke šeme baze podataka.

Specifikacija ograničenja domena (CREATE DOMAIN), za razliku od standarda, nije podržana kod Oracle Servera. Specifikacija ograničenja domena se obezbeđuje CHECK ograničenjem koje limitira dozvoljene vrednosti kolone tabele. Pored toga, postoji mogućnost za kreiranje složenih tipova podataka i to:

- objektnih tipova (*object_type*) i
- tipova ugnježdenih tabela (*nested_table_type*).

Oracle Server potpuno po standardu podržava SQL naredbe za kreiranje šema relacije (CREATE TABLE) i pogleda (CREATE VIEW), dok je naredba za izmenu šeme relacije podržana parcijalno. U svojoj sintaksi za izmenu definicije tabele, Oracle Server zahteva obavezno navođenje ključne reči COLUMN kod ADD i DROP klauzule, za razliku od standarda, kod kojeg je ova ključna reč opciona. Takođe, umesto standardne ključne reči ALTER, koristi ključnu reč MODIFY.

Kod specifikacije ograničenja FOREIGN KEY za operaciju brisanja, Oracle Server podržava izbor aktivnosti NO ACTION/RESTRICT, SET NULL i CASCADE, za razliku od standarda koji podržava još i SET DEFAULT. Za operaciju modifikacije, nije deklarativno podržana ni jedna od aktivnosti osim RESTRICT. Kod specifikacije ovog ograničenja, SQL ne podržava ni klauzulu MATCH, tj. ne dozvoljava ni delimično ni i potpuno referenciranje, već samo podrazumevano.

Kod brisanja tabele, podržane akcije su RESTRICT (podrazumevana vrednost) i CASCADE. Ove aktivnosti se odnose samo na ograničenja, za razliku od standarda koji vodi računa o referenciranju na brisanu tabelu ne samo za ograničenja, već i za poglede, SQL rutine i okidače. Kada se navodi ključna reč CASCADE, obavezno je i navođenje ključne reči CONSTRAINTS, što kod standarda nije slučaj.

Oracle Server ne podržava interne, već samo eksterne generatore sekvenci (sekvencere). Kod eksternih sekvencera, parametri se malo razlikuju u odnosu na standard. Oracle Server generator sekvenci ima keš i esplicitno definisano sortiranje (*order*), za razliku od standarda koji prepoznaje redosled sortiranja u zavisnosti od predznaka broja za inkrementiranje. Kod Oracle Servera se uvek mora definisati početna vrednost, za razliku od standarda čija početna vrednost ne mora biti eksplicitno navedena ako je naveden minimum ili maksimum i redosled sortiranja.

Sintaksa SQL naredbi SELECT, INSERT, UPDATE, i DELETE, kao i naredbi za deklarisanje kursora, potpuno su po standardu.

Sintaksa SQL naredbi za deklarisanje procedura i funkcija CREATE PROCEDURE i CREATE FUNCTION razlikuje se od standarda u sledećem:

- u standardu tip parametra (IN, OUT ili INOUT) se navodi pre imena parametara, a kod Oracle Servera posle;
- standard koristi oblik INOUT, dok Oracle Server IN OUT;
- Oracle Server zahteva ključnu reč AS ili IS nakon tipa podatka povratne vrednosti funkcije a pre definicije njenog tela, dok standard ovo izostavlja;
- Oracle Server omogućava da se telo rutine specificira pomoću jezika PL/SQL.

Sintaksna opcija OR REPLACE ne postoji u standardu, a služi za direktnu izmenu specifikacije procedure ili funkcije, bez prethodnog brisanja, ukoliko ona već postoji u rečniku podataka.

Sintaksa naredbe za definisanje okidača CREATE TRIGGER, u klauzuli REFERENCING, za razliku od standarda, ne podržava opcionu ključnu reč ROW, dok opcija za referenciranje TABLE nije uopšte podržana. Sintaksna opcija OR REPLACE postoji takođe i kod okidača, sa istom namenom kao i u slučaju drugih SQL naredbi. Oracle Server podržava i okidače tipa INSTEAD OF, koji standardom nisu predviđeni.

### 3. MEHANIZMI ZA REALIZACIJU OGRANIČENJA ZA MS SQL SERVER 2000/2005

Kada je u pitanju implementacija strukture i ograničenja baze podataka, MS SQL Server (verzija 2000 i 2005), pored SQL-a, predviđa i upotrebu proceduralno orijentisanog jezika koji se zove Transact-SQL (T-SQL). U ovom delu rada, diskutuje se usaglašenost koncepata koje podržava MS SQL Server sa standardom ANSI SQL:2003, u delu koji se odnosi na implementaciju logičke šeme baze podataka.

Kao i kod Oracle Servera, ni kod ovog SUBP-a kreiranje domena nije podržano kako to standard propisuje. Specifikacija domena se takođe obezbeđuje CHECK ograničenjem koje limitira dozvoljene vrednosti kolone tabele.

Kreiranje korisnički definisanih tipova podataka zasnovano je na sistemskim tipovima koje podržava SQL Server. Korisnički definisani tip može biti korišćen kada različite kolone tabela moraju imati iste tipove podataka, odnosno kada mora biti obezbeđeno da te kolone imaju isti tip podatka, dužinu i osobinu nulabilnosti (*nullability*), kojom se specificira da li tip podatka dozvoljava *null* vrednost. Korisnički tip podatka se kreira putem funkcije čiji je naziv *sp_addtype*. Ukoliko se korisnički definisani tip podatka specificira u bazi podataka **model,** koja se isporučuje u okviru SUBP-a, tada će tip podatka postojati u svim korisnički kreiranim bazama podataka. Ukoliko je tip kreiran u korisnički definisanoj bazi podataka, važiće samo u njenom okviru.

U sintaksi naredbe za izmenu šeme relacije, kod izmene definicije kolone, ključna reč COLUMN je obavezna iza klauzula ALTER i DROP, dok je iza ADD uvek izostavljena. Po standardu, ključna reč COLUMN uvek je opciona.

Sintaksa za definiciju kolone sadrži klauzulu IDENTITY, kao i kod ANSI standarda. IDENTITY označava interni sekvencer, samo što kod standarda ključnoj reči IDENTITY prethodi i konstrukcija GENERATED {ALWAYS | BY DEFAULT} AS. Za razliku od standarda, ovde se navode samo dva parametra: početna vrednost (*seed*) i korak za inkrement (*increment*). Ako ovi parametri nisu definisani, njihova predefinisana vrednost je (1, 1). T-SQL ne podržava eksterne generatore sekvenci.

Sintaksa SQL klauzule za deklarisanje ograničenja kolone u standardu podrazumeva da ako nije navedena klauzula NOT NULL, obeležje sme imati nula vrednost. Po sintaksi MS SQL-a, međutim, mora se navoditi ključna reč NOT NULL ili NULL za svako obeležje, jer ni jedna od ove dve klauzule nije podrazumevana.

U slučaju FOREIGN KEY klauzule za specifikaciju ograničenja stranog ključa, za operaciju brisanja i izmene, SQL Server podržava izbor aktivnosti NO ACTION/RESTRICT i CASCADE, za razliku od standarda koji podržava još SET NULL i SET DEFAULT.

Za razliku od Oracle Servera i standarda, MS SQL Server nema odloženu proveru ograničenja. Kao i Oracle Server, ni MS SQL Server ne podržava MATCH klauzulu, tj. ne dozvoljava delimično i potpuno referenciranje, već samo, podrazumevano.

SQL sintaksa za deklarisanje procedura i funkcija razlikuje se od standarda u sledećem:

- tip parametra (IN, OUT ili INOUT) u standardu se navodi pre imena parametra, dok se ovde uopšte ne navodi;
- kao i kod Oracle Servera, zahteva se ključna reč AS nakon tipa podatka povratne vrednosti funkcije, a pre definicije njenog tela, dok standard ovo izostavlja;
- SQL Server omogućava da se telo rutine specificira pomoću jezika jezika T-SQL.

Za razliku od standarda, SQL Server ne podržava pokretanje okidača za svaku torku (klauzula FOR EACH ROW), već samo pokretanje na nivou iskaza (to su tzv. *statement level* okidači). Takođe, nije podržana ni klauzula WHEN, kojom se definiše uslov za pokretanje okidača. Za implementaciju ove osobine, SQL Server obezbeđuje funkciju UPDATE(*column*), kojom se omogućava provera da li je menjana vrednost kolone tabele. MS SQL Server ne podržava u definiciji okidača ni opciju BEFORE, kojom se specificira trenutak okidanja neposredno pre izvršenja operacije. Podržani su tzv. AFTER okidači, kojima se specificira okidanje trigera neposredno nakon izvršenja operacije. Kao i u slučaju Oracle Servera, podržani su i okidači tipa INSTEAD OF.

Klauzula REFERENCING za definisanje *alias* liste starih i novih vrednosti uopšte nije podržana. Kod MS Server SQL-a, ne mogu se koristiti OLD i NEW promenljive. Umesto toga, postoje dve sistemske tabele, *Inserted* i *Deleted*, u kojima se čuvaju nove i stare vrednosti torki koje se upisuju ili brišu.

Sintaksa SQL naredbi SELECT, UPDATE i DELETE, kao i sintaksa naredbi za deklaraciju kursora, potpuno su po standardu. U sintaksi INSERT naredbe, za razliku od standarda, ključna reč INTO je opciona.

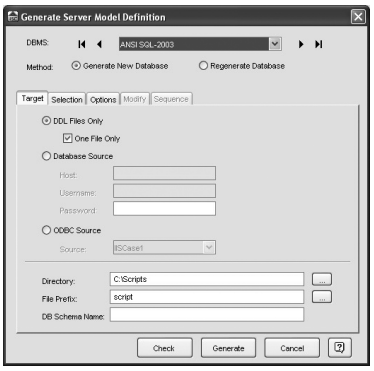
## 4. GENERISANJE OPISA ŠEME BAZE PODATAKA POMOĆU SQL GENERATORA

Polazna specifikacija za SQL generator, koja se dobija kao rezultat projektovanja u IIS*Case-u, jeste šema baze podataka u trećoj normalnoj formi sa definisanim skupovima svih ključeva šema relacija, kao i ograničenja: nula vrednosti, jedinstvenosti vrednosti, referencijalnog i inverznog referencijalnog integriteta. Ovi podaci smešteni su u rečniku podataka IIS*Case-a. Opis rečnika podataka dat je u [13].

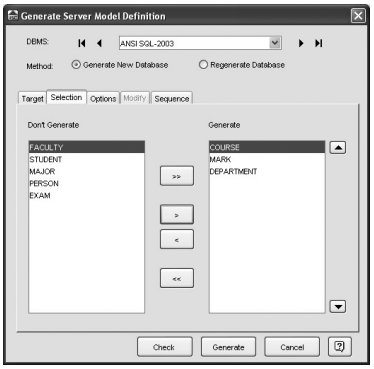
SQL generator obezbeđuje generisanje SQL opisa nove šeme baze podataka, ili modifikaciju postojeće, na tri načina:

- generisanjem skripta koji se kasnije izvršava na odgovarajućem SUBP-u,
- direktnim izvršavanjem SQL skripta na konektovanoj bazi podataka, ili
- direktnim izvršavanjem SQL skripta na konektovanom izvoru podataka, korišćenjem ODBC drajvera.

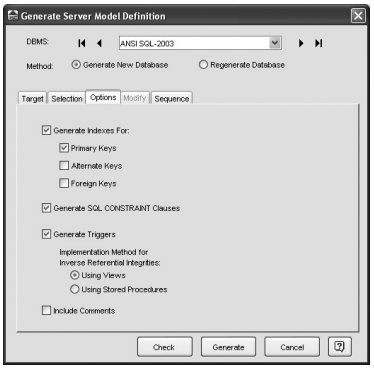
U sva tri slučaja generisani SQL kod memoriše se u skript datoteci. Na slikama od 1-5. prikazane su ekranske forme za definisaje vrednosti ulaznih parametara SQL generatora, dok je na slici 6. data ekranska forma za prikaz generisanog skripta.



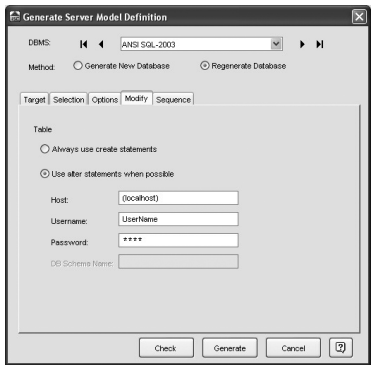

**Slika 1.** – *Prikaz stranice Target*



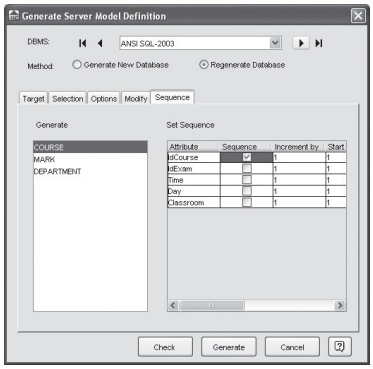

**Slika 2.** – *Prikaz stranice Selection*



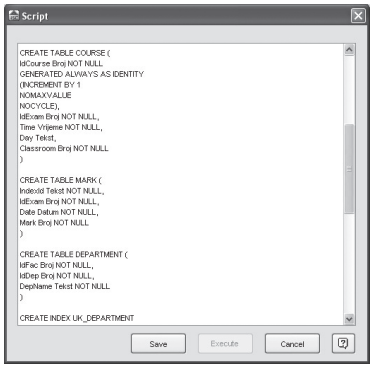

**Slika 3.** – *Prikaz stranice Option*




**Slika 4.** – *Prikaz stranice Modify*




**Slika 5.** – *Prikaz stranice Sequence*




**Slika 6.** – *Forma za prikaz skripta*

SQL generator podržava implementaciju sledećih tipova ograničenja: ograničenja domena, ograničenja ključa, ograničenja jedinstvenosti vrednosti obeležja, osnovnih ograničenja referencijalnih integriteta (podrazumevanih, parcijalnih i potpunih), proširenih ograničenja referencijalnih integriteta (podrazumevanih, parcijalnih i potpunih), ograničenja referencijalnih integriteta koji su posledica netrivijalnih zavisnosti sadržavanja (podrazumevanih, parcijalnih i potpunih), osnovna ograničenja inverznih referencijalnih integriteta i ograničenja inverznih referencijalnih integriteta koji su posledica netrivijalnih zavisnosti sadržavanja. [1, 3, 4, 8, 9]

Tip ograničenja kao i izbor aktivnosti za očuvanje konzistentnosti pri pokušaju narušavanja ograničenja (NO-ACTION, CASCADE, SET NULL, SET DEFAULT) kod dodavanja, modifikacije i brisanja torke, prepušta se projektantu. SQL generator uzima u obzir sve moguće relevantne kombinacije, koje projektant izabere, i implementira ograničenje. Ukoliko izabere akciju koja nije primenljiva u datoj situaciji, generator će dati upozorenje.

Ograničenja se implementiraju putem deklarativnih mehanizama SUBP, ukoliko za to postoji mogućnost, a u suprotnom putem proceduralnih. Za razliku od ANSI SQL:2003, komercijalni SUBP skromnije podržavaju deklarativno specificiranje međurelacionih ograničenja, zbog čega SQL generator veliki broj ograničenja implementira pomoću proceduralnih mehanizama.

## 5. PRIMERI IMPLEMENACIJE OGRANIČENJA

Razlike između MS SQL Servera i Oracle Servera ilustrovane su na primeru realizacije okidača za upravljanje brisanjem torke iz relacije $r(N_j)$ putem osnovnog parcijalnog referencijalnog integriteta $N_i[X] \subseteq_p N_j[Y]$. Za primer je uzeta realizacija okidača za upravljanje brisanjem torke iz referencirane relacije, za ograničenje osnovnog referencijalnog integriteta, jer je to jedan od najčešćih slučajeva koji se javljaju u praksi. Parcijalni tip ograničenja je izabran jer je njegova realizacija najsloženija. Implementacija ovog tipa ograničenja se veoma razlikuje za ova dva servera, te je zato i odabrana za prikaz u radu.

Zadatak algoritma okidača je da proveri da li, u relaciji $r(N_i)$, postoji torka $u$, koja referencira samo torku $v \in r(N_j)$. Tek ako takva torka $u$ postoji, okidač se pokreće. U suprotnom, $v$ se može brisati iz $r(N_j)$, bez obzira na specificiranu aktivnost ograničenja.

Primenom koncepata na kojima su zasnovani konkretni SUBP, SQL generator IIS*Case-a generiše SQL kod okidača za sledeći primer skupa šema relacija

***S* = {*Radnik*({MBR, IME, PREZIME, OOJ, RRM}, {MBR}), *RadnoMesto*({OOJ, RRM, NRM}, {OOJ+RRM})} i ograničenja referencijalnog integriteta *Radnik*[(OOJ, RRM)] ⊆ *RadnoMesto*[(OOJ, RRM)],**

koji predstavljaju model dela realnog sistema. Obeležja šeme relacije *RadnoMesto* imaju sledeća značenja: OOJ je "oznaka organizacione jedinice", RRM je "redni broj radnog mesta u organizacionoj jednici", a NRM je "naziv radnog mesta u organizacionoj jednici". Obeležja šeme relacije *Radnik* imaju sledeća značenja: MBR je "matični broj radnika", IME je "ime radnika", a PREZIME je "prezime radnika". Neka je za obeležja OOJ i RRM u opisu šeme relacije *Radnik* deklarisano da mogu imati nula vrednosti. Svaka nula vrednost ovih obeležja ukazuje da je radnik neraspoređen.

```
CREATE TRIGGER TRG_Radnik_RMesto_DEL
   ON RadnoMesto FOR DELETE
AS
 DECLARE @RRMR INTEGER(10),
         @OOJR INTEGER(10),
         @RRML INTEGER(10),
         @OOJL INTEGER(10)
 DECLARE Cursor_RadnoMesto CURSOR
 FOR SELECT RRM, OOJ FROM Deleted
 OPEN Cursor_RadnoMesto
 FETCH NEXT FROM Cursor_RadnoMesto
 INTO @RRMR, @OOJR
 WHILE @@FETCH_STATUS=0
 BEGIN
   DECLARE Cursor_Radnik CURSOR FOR
   SELECT RRM, OOJ FROM Radnik
   WHERE (RRM=@RRMR OR RRM IS NULL)
   AND (OOJ=@OOJR OR OOJ IS NULL)
   OPEN Cursor_Radnik
   FETCH NEXT FROM Cursor_Radnik
   INTO @RRML, @COJL
   WHILE @@FETCH_STATUS=0
   BEGIN
     IF NOT (@RRML IS NULL AND @OOJL IS NULL )
     BEGIN
      IF dbo.SadrzavanjePRI_Radnik(@RRML, @OOJL)=0
            <Izvrši aktivnost>
      END
      FETCH NEXT FROM Cursor_Radnik
      INTO @RRML, @OOJL
     END
     CLOSE Cursor_Radnik
     DEALLOCATE Cursor_Radnik
     FETCH NEXT FROM Cursor_RadnoMesto
     INTO @RRMR, @OOJR
   END
   CLOSE Cursor_RadnoMesto
   DEALLOCATE Cursor_RadnoMesto
```

**Slika 7.** – *Okidač za upravljanje brisanjem torke, za MS SQL Server*

U nastavku rada, biće paralelno prikazivane određene funkcionalnosti za MS SQL Server i Oracle Server, radi lakšeg poređenja rešenja za oba servera. Na slici 7. prikazana je realizacija okidača za upravljanje brisanjem, za MS SQL Server, dok je na slikama 8, 10. i 11. prikazano isto rešenje za Oracle Server.

Pošto MS SQL Server u specifikaciji okidača nema opciju okidanja za svaku torku, niti opciju kojom se specificira trenutak okidanja neposredno pre izvršenja operacije, za realizaciju brisanja se moraju koristiti ugnježđeni kursori. U prvi kursor (*Cursor_RadnoMesto*) smeštaju se sve torke iz tabele *Deleted*. *Deleted* je sistemska tebela u kojoj se nalaze sve obrisane torke iz relacije $r(N_j)$. Ugnježđeni kursor (*Cursor_Radnik*) rezervisan je za sve torke koje referenciraju odgovarajuću torku iz spoljašnjeg kursora.

Oracle Server podržava pokretanje okidača za svaku torku (FOR EACH ROW), kao i specifikaciju trenutka izvršenja okidača BEFORE i AFTER. Ovaj SUBP, takođe, podržava definisanje globalnih promenljivih i kreiranje javnih paketa. Ove osobine su iskorišćene u realizaciji okidača za upravljanje brisanjem, čija implementacija se znatno razlikuje od implementacije istog okidača kod MS SQL Servera. Saglasno tome, u slučaju brisanja torke iz relacije $r(N_j)$ putem ograničenja parcijalnog referencijalnog integriteta $N_i[X] \subseteq_p N_j[Y]$ za Oracle Server formira se nekoliko okidača i procedura.

Prvi okidač (slika 8) pokreće se na nivou iskaza, pre brisanja torke iz relacije $r(N_j)$ i ima zadatak da inicijalizuje pomoćne strukture podataka, koje će se koristiti u okviru sledeća dva okidača. Pomoćne strukture deklarišu se unutar jednog paketa, nazvanog *RI_Radnik_RadnoMesto_PCK* (slika 9), koji se posebno kreira za odgovarajuće ograničenje.

```
CREATE OR REPLACE TRIGGER TRG_Radnik_RMestoRD1 BEFORE DELETE
ON RadnoMesto
BEGIN
 RI_Radnik_RadnoMesto_PCK.Del_Count := 0;
 RI_Radnik_RadnoMesto_PCK.For_Del_RadnoMesto.DELETE;
END;
```

**Slika 8.** – *Prvi okidač za upravljanje brisanjem torke, za Oracle Server*

```
CREATE OR REPLACE PACKAGE RI_Radnik_RadnoMesto_PCK
IS
  TYPE TRecDelRadnoMesto IS
  RECORD (RRM RadnoMesto.RRM%TYPE, OOJ RadnoMesto.OOJ%TYPE);
  TYPE TTabForDelete IS TABLE OF TrecDelRadnoMesto
  INDEX BY BINARY_INTEGER;
  For_Del_RadnoMesto RI_Radnik_RadnoMesto_PCK.TTabForDelete;
  Del_Count NUMBER(8,0);
END;
```

**Slika 9.** – *PL/SQL kôd paketa RI_Radnik_RadnoMesto_PCK, za Oracle Server*

Struktura *For_Del_RadnoMesto* (slika 9) služi za prenos vrednosti obeležja iz *Y*, svih torki koje treba obrisati. Ova struktura ima ulogu kao i tabela *Deleted* kod MS SQL Servera. U promenljivoj *Del_Count* se čuva broj torki koje treba da budu obrisane.

Drugi okidač (slika 10), koji se pokreće neposredno pre izvršavanja brisanja torke iz relacije $r(N_j)$, ima zadatak da u pomoćne strukture smesti vrednosti obeležja iz *Y*, torke koja se briše. Pri tome vrednosti obeležja iz *Y* predstavljaju vrednost ključa torke koja se briše.

```
CREATE OR REPLACE TRIGGER TRG_Radnik_RMestoRD2
BEFORE DELETE ON RadnoMesto
FOR EACH ROW
DECLARE
  v RadnoMesto%ROWTYPE;
BEGIN
 v.RRM := :OLD.RRM;
 v.OOJ := :OLD.OOJ;
 IF Global_PCK.SadrzavanjePRI_RadnoMesto(v) THEN
  RI_Radnik_RadnoMesto_PCK.Del_Count :=
  RI_Radnik_RadnoMesto_PCK.Del_Count + 1;
  RI_Radnik_RadnoMesto_PCK.For_Del_RadnoMesto
   (RI_Radnik_RadnoMesto_PCK.Del_Count).RRM := v.RRM;
  RI_Radnik_RadnoMesto_PCK.For_Del_RadnoMesto
   (RI_Radnik_RadnoMesto_PCK.Del_Count).OOJ := v.OOJ;
 END IF;
END;
```

**Slika 10.** – *Drugi okidač za upravljanje brisanjem torke, za Oracle Server*

Treći okidač (slika 11) pokreće se na nivou iskaza, nakon sprovedenog brisanja iz relacije $r(N_j)$ i koristi pomoćne strukture podataka, inicijalizovane putem drugog okidača. U ovom okidaču koristi se samo jedan kursor (*Cursor_Radnik*).

```
CREATE OR REPLACE TRIGGER TRG_Radnik_RMestoRD3
AFTER DELETE ON RadnoMesto
DECLARE
  t RadnoMesto%ROWTYPE;
  u Radnik%ROWTYPE;
  CURSOR Cursor_Radnik( C_RRM RadnoMesto.RRM%TYPE,
      C_OOJ RadnoMesto.OOJ%TYPE) IS SELECT *
      FROM Radnik WHERE
    (Radnik.RRM = C_RRM OR Radnik.RRM IS NULL) AND
    (Radnik.OOJ = C_OOJ OR Radnik.OOJ IS NULL);
BEGIN
  FOR i IN 1..RI_Radnik_RadnoMesto_PCK.Del_Count
  LOOP
   t.RRM := RI_Radnik_RadnoMesto_PCK.For_Del_RadnoMesto
     (RI_Radnik_RadnoMesto_PCK.Del_Count).RRM;
   t.OOJ := RI_Radnik_RadnoMesto_PCK.For_Del_RadnoMesto
     (RI_Radnik_RadnoMesto_PCK.Del_Count).OOJ;
    OPEN Cursor_Radnik(t.RRM, t.OOJ);
    LOOP
      FETCH Cursor_Radnik INTO u;
      EXIT WHEN Cursor_Radnik%NOTFOUND;
      IF NOT Global_PCK.SadrzavanjePRI_Radnik(u) THEN
          <Izvrši aktivnost>
      END IF;
    END LOOP;
    CLOSE Cursor_Radnik;
  END LOOP;
END;
```

**Slika 11.** – *Treći okidač za upravljanje brisanjem torke, za Oracle Server*

```
CREATE PROCEDURE Restrict
AS
 RAISERROR('Torka u ne moze da se briše iz relacije ', 16,1)
 ROLLBACK TRAN
```

**Slika 12.** – *Procedura za sprečavanje brisanja, za MS SQL Server*

```
CREATE PROCEDURE Restrict
IS
  exc EXCEPTION;
BEGIN
  RAISE exc;
  EXCEPTION
  WHEN exc THEN RAISE_APPLICATION_ERROR
   (-20000,'Torka u ne moze da se brise iz relacije!');
END;
```

**Slika 13.** – *Procedura za sprečavanje brisanja, za Oracle Server*

Pseudo naredba <Izvrši aktivnost>, koja se pojavljuje u okidaču MS SQL Servera (slika 7) i u trećem okidaču Oracle Servera (slika 11), zamenjuje se pozivom jedne od

```
CREATE PROCEDURE SetNull(@MBR INTEGER(10))
AS
  UPDATE Radnik SET RRM = NULL, OOJ = NULL WHERE (MBR = @MBR)
```

**Slika 14.** – *Procedura SetNull, za MS SQL Server*

```
CREATE PROCEDURE SetNull(u IN Radnik%ROWTYPE)
IS
BEGIN
  UPDATE Radnik SET RRM = NULL, OOJ = NULL
  WHERE (MBR = u.MBR);
END;
```

**Slika 15.** – *Procedura SetNull, za Oracle Server*

```
CREATE PROCEDURE KaskadnoDelPRI(@MBR INTEGER(10))
AS
  DELETE FROM Radnik WHERE (MBR = @MBR)
```

**Slika 16.** – *Procedura KaskadnoDelPRI, za MS SQL Server*

```
CREATE PROCEDURE KaskadnoDelPRI(u IN Radnik%ROWTYPE)
IS
BEGIN
  DELETE FROM Radnik WHERE (MBR = u.MBR);
END;
```

**Slika 17.** – *Procedura KaskadnoDelPRI, za Oracle Server*

```
CREATE FUNCTION SadrzavanjePRI_Radnik(@RRM INTEGER(10),
                                      @OOJ INTEGER(10))
RETURNS int
AS
BEGIN
  DECLARE
      @Count int,
      @Ret  int
  SELECT @Count = COUNT(*) FROM RadnoMesto v
  WHERE (@RRM IS NULL OR v.RRM = @RRM) AND
        (@OOJ IS NULL OR v.OOJ = @OOJ)
  IF @Count != 0
      SET @ret=1
  ELSE
      SET @ret=0
  RETURN @ret
END
```

**Slika 18.** – *Funkcija SadrzavanjePRI_Radnik, za MS SQL Server*

```
CREATE FUNCTION SadrzavanjePRI_Radnik(u IN Radnik%ROWTYPE)
RETURN BOOLEAN
IS
   I NUMBER;
BEGIN
  SELECT COUNT(*) INTO I FROM RadnoMesto v
  WHERE (u.RRM IS NULL OR v.RRM = u.RRM) AND
  (u.OOJ IS NULL OR v.OOJ = u.OOJ);
  IF I <> 0 THEN
      RETURN TRUE;
  ELSE
      RETURN FALSE;
  END IF;
END;
```

**Slika 19.** – *Funkcija SadrzavanjePRI_Radnik, za Oracle Server*

sledećih procedura: *Restrict*, *SetNull*, *Cascade* ili *SetDefault*, u zavisnosti od izabrane aktivnosti koja treba da se sprovodi pri brisanju, ukoliko dođe do narušavanja integriteta baze podatka. Na slici 12. prikazana je procedura za sprečavanje brisanja torke, za MS SQL Server, koja odgovara istoj takvoj proceduri za Oracle Server, datoj na slici 13.

Na slici 14. prikazana je procedura brisanja torke svođenjem na nula vrednost, za MS SQL Server, koja odgovara istoj takvoj proceduri za Oracle Server, datoj na slici 15.

Na slici 16. prikazana je procedura za kaskadno brisanje torke, za MS SQL Server, koja odgovara istoj takvoj proceduri za Oracle Server, datoj na slici 17.

Ako se uporedi SQL kôd okidača SQL Servera i Oracle Servera, može se uočiti da je PL/SQL-om prenos parametara kraći i jednostavniji. Tome doprinosi postojanje tipa podataka ROWTYPE koji obezbeđuje deklaraciju samo jedne promenljive *u* tog tipa, dok se T-SQL-om mora deklarisati onoliko promenljivih koliko ima obeležja u primarnom ključu šeme relacije Radnik.

Na slikama 18. i 19. prikazana je implementacija funkcije *SadrzavanjePRI_ Radnik*, za MS SQL Server i Oracle Server, respektivno. Pošto MS SQL Server ne podržava logički tip podatka, njegova funkcija vraća različiti tip podatka od iste funkcije, realizovane za Oracle Server.

Procedure za brisanje torke svođenjem na predefinisanu (*default*) vrednost mogu se videti u [1].

## 6. POREĐENJE KARAKTERISTIKA MS SQL SERVERA I ORACLE SERVERA

U narednoj tački biće data kratka komparacija mogućnosti MS SQL Servera i T-SQL-a, s jedne strane, i Oracle Servera i PL/SQL-a, s druge. Tabela 1. daje sumarni pregled poređenja ova dva servera i jezika, a u nastavku, daje se i tekstualni opis ovih razlika. Reč je o razlikama koje imaju bitan uticaj na izbor načina implementacije ograničenja baze podataka.

Kod MS SQL Servera ne postoje paketi tj. kolekcije procedura i funkcija. Takođe, nije podržana deklaracija globalnih promenljivih.

Vreme okidanja trigera može biti samo AFTER, a ne postoji BEFORE kao kod Oracle Servera. Takođe, SQL Server ne podržava okidače na nivou torke, već samo na nivou iskaza.

SQL Server nema objektnih tipova podataka kao Oracle Server, dok je definisanje korisničkih tipova podataka veoma oskudno. Kod MS SQL Servera nisu podržani eksterni generatori sekvenci, dok Oracle Server nema interne sekvencere, tj. osobinu IDENTITY za kolone tabela.

MS SQL Server ne podržava logički tip podatka (*boolean*), niti tip podatka ROWTYPE.

MS SQL Server, za razliku od Oracle Servera, takođe ne podržava ni odlaganje trenutka provere ograničenja.

Pri operaciji brisanja torke iz relacije, aktivnosti za očuvanje konzistentnosti baze podatka kod SQL Servera (one koje se definišu deklarativnim putem) mogu biti *no action* i *cascade,* dok kod Oracle Servera može se definisati i *not null*. Pri operciji modifikacije torke, SQL Server opet podržava *no action* i *cascade,* dok Oracle ne podržava ni jednu aktivnost osim *no action* koja je data kao predefinisana.

Za izmenu definicije tabele, okidača, procedure i ostalih objekata baze podataka, kod MS SQL Servera mora uvek da se proverava da li tabela postoji i tek ako postoji, da se izvrši komanda ALTER, dok kod Oracle Servera postoji opcija da se kod definicije nekih objekata baze podataka (npr. procedura,

| Osobine | | Oracle | MS SQL Server |
|---|---|---|---|
| Tabele | | Relacione tabele,<br>Objektne tabele | Relacione tabele |
| Okidači | | BEFORE okidači,<br>AFTER okidači,<br>INSTEAD OF okidači | AFTER okidači,<br>INSTEAD OF okidači |
| | | ROW ,<br>STATEMENT | STATEMENT |
| Aktivnosti | ON DELETE | NO-ACTION, CASCADE,<br>SET NULL | NO-ACTION, CASCADE |
| | ON UPDATE | NO-ACTION | NO-ACTION, CASCADE |
| Procedure | | PL/SQL izrazi,<br>Java metode,<br>(3GL) rutine | T-SQL izrazi |
| Nizovi | | Podržano | Nije podržano |
| Objektni tipovi podataka | | Podržano | Nije podržano |
| Tip podatka ROWTYPE | | Podržano | Nije podržano |
| Boolean tip podatka | | Podržano | Nije podržano |
| Složeni tipovi podataka | | Podržano | Nije podržano |
| Paketi | | Podržano | Nije podržano |
| Odlaganje trenutka provere ograničenja | | Podržano | Nije podržano |
| Eksterni sekvenceri | | Podržano | Nije podržano |
| Interni sekvenceri | | Nije podržano | Podržano |
| User-defined type | | Podržano | Oskudno podržano |
| Deklaracija globalnih promenljivih | | Podržano | Nije podržano |

**Tabela 1.** – *Sumarni rezultati poređenja PL/SQL-a i T-SQL-a*

funkcija i okidača) koristi opcija OR REPLACE. Korišćenjem ovakve opcije, nije potrebno eksplicitno proveravanje postojanja objekta u rečniku baze podataka.

T-SQL i PL/SQL imaju različite mehanizme za obradu greške. Kod T-SQL-a koristi se sistemski definisana promenljiva @@error. Ona se testira eksplicitno nakon svake SQL naredbe koja može prouzrokovati grešku. Kod PL/SQL-a, u proceduri se deklariše sekcija EXCEPTION. Kad god se dogodi greška, Oracle Server prebacuje tok izvršenja programa na ovu sekciju. (Slika 13 ilustruje jedan takav primer).

Sve potrebne funkcionalnosti SQL generatora uspešno se realizovane, za oba izabrana servera. Na osnovu iskustava autora stečenih pri realizaciji SQL generatora, kao i samo na osnovu ove analize, teško je suditi o prednostima i manama Oracle Servera i MS SQL Servera, a posebno o tome da li favorizovati jedan ili drugi server, kada je u pitanju lakoća pisanja koda za istu funkcionalnost. To pitanje u mnogome može biti opredeljeno znanjem, nivoom obučenosti i privrženošću programera jednom ili drugom serveru. Odluka o tome da li se u praksi opredeliti za jedan ili drugi server zavisi kako od njihovih tehničkih mogućnosti, navedenih u tabeli 1, tako i od mnogih drugih faktora. Ocena onih karakteristika izabranih servera koje se odnose na načine i mogućnosti realizacije integritetne komponente šeme baze podataka, svakako zahteva detaljniju analizu koja izlazi iz okvira ovog rada.

## 7. ZAKLJUČAK

Cilj ovog rada bio je da se prikažu specifičnosti generisanja SQL opisa šeme baze podataka i poređenje dva izabrana SUBP-a: MS SQL Server (verzije 2000 i 2005) i Oracle Server (verzije 9i i 10g). Pokazano je, takođe, u kojoj meri ova dva SUBP-a podržavaju standard ANSI SQL:2003, kada je u pitanju implementacija relacione šeme baze podataka. Sve ovo je ilustrovano SQL kodom koji generiše SQL generator IIS*Case-a po sintaksi Microsoft SQL i T-SQL i Oracle SQL i PL/SQL . MS SQL Server 2000/2005 i Oracle Server 9i/10g su izabrani jer je smatrano da su široko rasprostranjeni u praksi. Dalji pravci razvoja usmereni su ka proširivanju funkcionalnosti SQL generatora alata IIS*Case opcijama generisanja SQL opisa šeme baze podataka za druge SUBP-ove, kao i opcijama podrške još nekih specijalnih tipova ograničenja, koji ovom verzijom nisu bili obuhvaćeni.

## 8. LITERATURA


[1] Aleksić S, Jedan SQL generator implementacionog opisa šeme baze podataka CASE alata IIS*Case, Magistarski rad, Univerzitet u Novom Sadu, Fakultet tehničkih nauka, Novi Sad, 2006;

[2] ARTech. *DeKlarit*™ (*The Model-Driven Tool for Microsoft Visual Studio 2005*), Chicago, U.S.A. Available at: http://www.deklarit.com [September, 2007].

[3] Govedarica M, Generisanje skupa međurelacionih ograničenja implementacione šeme baze podataka", Magistarski rad, Univerzitet u Novom Sadu, Fakultet tehničkih nauka, Novi Sad 1998;

[4] ISO/IEC 9075-{1, 2, 11}:2003, (ANSI SQL:2003), American National Standards Institute, 2003;

[5] Luković I, Mogin P, Pavićević J, Ristic S, An Approach to Developing Complex Database Schemas Using Form Types, Software: Practice and Experience, John Wiley & Sons Inc, Hoboken, USA, ISSN: 0038-0644, Published Online, May 29, 2007, DOI: 10.1002/spe.820;

[6] Microsoft SQL Server 2000, User Manuals;

[7] Microsoft SQL Server 2005, User Manuals;

[8] Mogin P, Luković I, Govedarica M, Principi projektovanja baza podataka, FTN izdavaštvo, Novi Sad, 2004, ISBN: 86-80249-81-5;

[9] Mogin P, Luković I, Govedarica M, Extended Referential Integrity, Novi Sad Journal of Mathematics, Novi Sad, Yugoslavia, ISSN: 1450-5444, Vol. 30, No. 3, 2000, pp. 111-122.

[10] Oracle DBMS 9i, User Manuals;

[11] Oracle DBMS 10g, User Manuals;

[12] Oracle Designer 9i, On-line Documentation;

[13] Pavićević J, Razvoj CASE alata za automatizovano projektovanje i integraciju šema baza podataka, Magistarski rad, Univerzitet Crne Gore, Prirodno-matematički fakultet, Podgorica, 2005;

[14] Pavićević J, Luković I, Mogin P, Govedarica M, Information System Design and Prototyping Using Form Types, INSTICC I International Conference on Software and Data Technologies, Setubal, Portugal, September 11-14, 2006, Proceedings, Vol. 2, pp. 157-160;

[15] Aleksić S, Luković I, Mogin P, Govedarica M, A Generator of the SQL Specification of a Database Schema, International Multiconference on Computer Science and Information Technology, 1st Workshop on Advances in Programming Languages (WAPL'07), October 15-17, 2007, Wisla, Poland; (prihvaćen rad)



Mr Slavica Aleksić, asistent, Fakultet tehničkih nauka, Novi Sad, Departman za informatiku i računarstvo
Naučne oblasti: baze podataka, informacioni sistemi



Dr Ivan Luković, redovni profesor, Fakultet tehničkih nauka, Novi Sad, Departman za informatiku i računarstvo
Naučne oblasti: baze podataka, informacioni sistemi, softversko inženjerstvo