

Objektno-relacioni model



Osnovne karakteristike

- **Korisnički definisani tipovi**

- ☐ **Distinct tip**
- ☐ **Strukturirani tip**
- ☐ **Metode**

- **Konstruisani tipovi**

- ☐ **Referentni tipovi**
- ☐ **Tip vrsta**
- ☐ **Kolekcija**



Korisnički definisani tipovi

Distinkt tip

- Distinkt tip je jednostavan, perzistentni, imenovani korisnički definisani tip, čijim uvođenjem je podržano strogo tipiziranje.
- Distinkt tipovi su uvek konačni (FINAL), što znači da ne mogu imati podtipove, odnosno da za njih nije podržano nasleđivanje.
- Distinkt tip i njegov izvorni predefinisani tip nisu direktno uporedivi.

Korisnički definisani tipovi

Distinkt tip

Sintaksa definisanja:

```
CREATE TYPE <naziv distinkt tipa>  
    AS <predefinisani tip> FINAL  
    [<cast opcije prevodjenja>]
```

Primer:

```
CREATE TABLE Soba (SobaID CHAR(10),  
                    Duzina INTEGER,  
                    Sirina INTEGER,  
                    Povrsina INTEGER);  
  
UPDATE Soba  
SET Povrsina = Duzina; => OK
```

Korisnički definisani tipovi

Distinkt tip

SQL:1999 standard, uvodjenjem distinkt tipova, rešava prethodni problem:

CREATE TYPE identifikator_sobe AS CHAR(10) FINAL;

CREATE TYPE metar AS INTEGER FINAL;

CREATE TYPE kvadratni_metar AS INTEGER FINAL;

*CREATE TABLE Soba(SobaID identifikator_sobe,
 Duzina metar,
 Sirina metar,
 Povrsina kvadratni_metar);*

UPDATE Soba

SET Povrsina = Duzina; => GRESKA

Korisnički definisani tipovi

Distinkt tip

- *Kada su argumenti izraza kolone, definisane nad distinkt tipovima podataka, neophodno je transformisati njihove vrednosti u izvorne predefinisane tipove:*

*SELECT (Duzina + Sirina) * 2*

FROM Soba

WHERE SobaID = 'xxx'; => GRESKA

Ovaj upit neće moći da se izvrši - operatori sabiranja (+) i množenja () definisani su nad numeričkim tipovima podataka*

Korisnički definisani tipovi

Distinkt tip

- *Da bi izračunali obim sobe, upit treba modifikovati uključivanjem eksplicitne transformacije argumenata Duzina i Sirina u izvorne predefinisane tipove, korišćenjem funkcije CAST:*

```
SELECT (CAST (Duzina AS INTEGER) + CAST (Sirina AS INTEGER)) *  
      2  
FROM Soba  
WHERE SobaID = 'xxx';           => OK
```

**CAST (SOURCE AS DISTINCT) WITH <naziv distinkt tipa>
CAST (DISTINCT AS SOURCE) WITH <predefinisani tip>**

Korisnički definisani tipovi

Struktuirani tip

- Omogućuje definisanje perzistentnih, imenovanih tipova, koji mogu imati jedan ili više atributa.
- Atributi mogu biti bilo kog tipa, uključujući druge struktuirane tipove, nizove...

- Sintaksa:

CREATE TYPE <naziv struktuiranog tipa>

[UNDER <naziv nadtipa>]

AS (<naziv atributa> <tip atributa>, ...)

[[NOT] INSTANTIABLE]

NOT FINAL

[<specifikacija referenciranja>]

[<specifikacija metode>, ...]

Korisnički definisani tipovi

Strukturirani tip

- U sledećem primeru kreiraju se dva strukturana tipa: tip osoba, koji se ne instancira (nema svoja pojavljivanja) i koji može imati podtipove, i tip student, koji je podtip tipa osoba, instancira se i takodje može imati podtipove:

```
CREATE TYPE osoba AS  
(...) NOT INSTANTIABLE NOT FINAL
```

```
CREATE TYPE student UNDER osoba AS  
(...) NOT FINAL
```

Korisnički definisani tipovi

- *U sledećem primeru dopunjena je definicija struktuiranog tipa osoba specificiranjem njegove strukture, odnosno atributa prime, jmbg i pol:*

```
CREATE TYPE osoba AS (prime CHAR(30),  
                        jmbg  INTEGER,  
                        pol   CHAR(1))  
NOT INSTANTIABLE NOT FINAL;
```

Korisnički definisani tipovi

Metode

Originalne metode se specificiraju sledećom sintaksom:

[INSTANCE | STATIC]

METHOD <naziv metode>

(<naziv parametra> <tip parametra>, ...)

RETURNS <tip rezultata>

[SPECIFIC <specifični naziv>]

[SELF AS RESULT]

[SELF AS LOCATOR]

[<karakteristike metode>]

Korisnički definisani tipovi

Metode

Redefinisane metode se specificiraju sledećom sintaksom:

**OVERRIDING [INSTANCE | STATIC]
METHOD <naziv metode>
(<naziv parametra> <tip parametra>, ...)
RETURNS <tip rezultata>
[SPECIFIC <specifični naziv>]**

Korisnički definisani tipovi

Transformacija se specificira sledećim iskazom:

**{TRANSFORM | TRANSFORMS} FOR < naziv tipa >
<naziv transformacione grupe>
(TO SQL WITH <to sql funkcija>
| FROM SQL WITH <from sql funkcija>)**

Korisnički definisani tipovi

U primeru koji sledi kreira se struktuirani tip radnik sa strukturom koju čine atributi ime, osnovna_plata i bonus. Tip može imati pojavljivanja i može se specijalizovati u podtipove :

```
CREATE TYPE radnik AS(ime      CHAR(40),  
                        osnovna_plata DECIMAL(9,2),  
                        bonus      DECIMAL(9,2))  
INSTANTIABLE NOT FINAL  
METHOD plata() RETURNS DECIMAL(9,2);
```

Korisnički definisani tipovi

Nakon što je specificirana kao deo definicije tipa radnik, metoda plata() se kreira CREATE METHOD naredbom. Metoda mora biti kreirana u istoj šemi baze podataka kao i struktuirani tip za koji se definiše:

```
CREATE METHOD plata() FOR radnik  
BEGIN  
...  
END;
```

Korisnički definisani tipovi

Sledećom naredbom možemo kreirati novi struktuirani tip menadzer koji je podtip struktuiranog tipa radnik. On ima jedan dodatni atribut udeo, može imati pojavljivanja i može se dalje specijalizovati:

```
CREATE TYPE menadzer UNDER radnik AS(udeo INTEGER)
INSTANTIABLE NOT FINAL;
--redefinisana
OVERRIDING METHOD plata() RETURNS DECIMAL(9,2);
--originalna
METHOD zaduzenje() RETURNS INTEGER ;
```


Korisnički definisani tipovi

*Prilikom definisanja struktuiranog tipa sistem automatski generiše jednu **observer** i jednu **mutator** metodu za svaki atribut. One imaju isti naziv kao i atribut za koji su generisane, služe za pristup, odnosno izmenu vrednosti atributa i ne mogu se redefinisati.*

***Konstruktor** metoda ili konstruktor kreira novu instancu tipa i postavlja attribute na default vrednosti. Ova metoda ima isti naziv kao struktuirani tip i nema argumente, što znači da se za struktuirani tip ST referencira sa ST(), a povratni tip je ST*

Konstruktor se može pozivati sa odgovarajućim argumentima, čime se određuju inicijalne vrednosti atributa. Korisnički definisani konstruktori pozivaju se sa operatorom NEW:

NEW <ime_metode> <lista parametara>

Korisnički definisani tipovi

U sledećem primeru prvo definišemo korisnički tip adresa i korisnički konstruktor, a zatim kreiramo nove instance tipa adresa uz pomoć sistemskog i korisničkog konstruktora i pristupamo atributima uz pomoć observer i mutator metoda:

```
CREATE TYPE adresa AS
```

```
(ulica CHAR (30),
```

```
grad CHAR (20),
```

```
postbr INTEGER) NOT FINAL
```

```
METHOD adresa (ul CHAR (30), gr CHAR (20), pbr INTEGER)
```

```
RETURNS adresa
```

```
CREATE METHOD adresa (ul CHAR (30), gr CHAR (20), pbr INTEGER)
```

```
RETURNS adresa
```

```
BEGIN
```

```
SET self.ulica = ul;
```

```
SET self.grad = gr;
```

```
SET self.postbr = pbr;
```

```
RETURN adresa;
```

```
END;
```

Korisnički definisani tipovi

```
CREATE TABLE Adrese OF adresa;
```

```
BEGIN
```

```
    DECLARE adr1, adr2 adresa;
```

```
    SET    adr1    =    adresa().ulica('Futoska    45').grad('Novi  
    Sad').postbr(21000);
```

```
    SET adr2 = NEW adresa('Kosovska 17', 'Beograd', 11000);
```

```
SELECT ulica()
```

```
FROM Adrese
```

```
WHERE grad() = 'Beograd';
```

```
UPDATE Adrese
```

```
SET postbr = 11010
```

```
WHERE grad() = 'Beograd';
```

```
END;
```

Korisnički definisani tipovi

Tabele tipova

Ukoliko je tabela T definisana direktno nad struktuiranim tipom ST ona se naziva tabela tipa (*typed table*).

- Tabela tipa se koristi za uskladištenje instanci tipa
- Nad istim struktuiranim tipom može biti definisano više tabela
- Tabela tipa ima kolone koje po nazivu i tipu odgovaraju atributima struktuiranog tipa i jednu kolonu REFC koja je njena referentna kolona (*self-referencing*)
- Deklarisani tip kolone REFC je obavezno REF(ST), a njene vrednosti ne mogu biti NULL
- Deklaracija tabele tipa može se dopuniti elementima koji se ne preuzimaju od struktuiranog tipa, kao što je specifikacija primarnog ključa, spoljnih ključeva i ograničenja na nivou tabele

Korisnički definisani tipovi

Kreiranje tabele tipa:

CREATE TABLE <naziv tabele> OF <struktuirani tip>

Sledi primer kojim se pokazuje način definisanja tabele nad tipom. Prvo definišemo struktuirane tipove adresa, osoba i nekretnine.

CREATE TYPE adresa AS(...) NOT FINAL

CREATE TYPE osoba AS(...) NOT FINAL

**CREATE TYPE nekretnina AS (nekretninaID INTEGER
 opis VARCHAR(50),
 lokacija adresa,
 povrsina DECIMAL(8,2),
 vlasnik osoba) NOT FINAL**

CREATE TABLE vlasnistvo OF nekretnine;



Korisnički definisani tipovi

Hijerarhija i nasledjivanje

U SQL:1999 standardu je podržano jednostruko nasledjivanje i hijerarhija struktuiranih tipova (broj nivoa hijerarhije je proizvoljan).

- Vrednost struktuiranog tipa ST1 može se dodeliti promenljivoj struktuiranog tipa ST2 ako i samo ako je ST1 podtip od ST2.
- Struktuirani tip ST1 je direktni podtip tipa ST2 ako je ST1 podtip od ST2 i ne postoji tip ST3 koji je podtip od ST2 i nadtip od ST1.
- Tip koji nema nijedan odgovarajući nadtip naziva se maksimalni nadtip, a tip koji nema nijedan podtip naziva se tip list.
- Skup svih podtipova nekog maksimalnog nadtipa ST, koji mora biti jedinstven, naziva se familija podtipova od ST.

Korisnički definisani tipovi

Poredjenje korisnički definisanih tipova

- Objekti, odnosno instance nekog korisnički definisanog tipa su apstraktni
- Čak i u slučaju da su sve komponente dva objekta identične, oni se neće smatrati jednakim sve dok se na neki način ne kaže sistemu da ih tretira kao jednake
- Ne možemo koristiti ORDER BY klauzulu niti poredjenje tipa "<" u WHERE klauzuli ukoliko nismo u mogućnosti da poredimo bilo koja dva elementa

Korisnički definisani tipovi

Da bi se omogućilo poredjenje i sortiranje objekata struktuiranih tipova u SQL:1999 standardu uvedena je CREATE ORDERING naredba:

**CREATE ORDERING FOR T
EQUALS ONLY BY STATE;**

Sledeći oblik CREATE ORDERING naredbe omogućuje primenu svih operatora poredjenja (<, <=, >, >=, = i <>) na objekte struktuiranog tipa T:

**CREATE ORDERING FOR T
ORDERING FULL BY RELATIVE WITH F;**

Korisnički definisani tipovi

- *Da bi definisali kako se objekti x_1 i x_2 tipa T porede uvodi se funkcija F čiji su argument objekti x_1 i x_2 .*
- *Funkcija F se mora napisati tako da je $F(x_1, x_2) < 0$ kad god želimo da zaključimo da je $x_1 < x_2$. $F(x_1, x_2) = 0$ označava da je $x_1 = x_2$, a $F(x_1, x_2) > 0$ označava da je $x_1 > x_2$.*
- *Ukoliko "ORDERING FULL" zamenimo sa "EQUALS ONLY" funkcija $F(x_1, x_2) = 0$ označava da je $x_1 = x_2$, dok sve ostale vrednosti funkcije $F(x_1, x_2)$ označavaju da je $x_1 \neq x_2$.*
- *Poredjenje u odnosu na operator "<" je nemoguće u tom slučaju.*

Korisnički definisani tipovi

- *Promena definicije i izbacivanje korisnički definisanog tipa*

ALTER TYPE <naziv tipa> <akcija promene>

Akcija promene može biti jedna od sledećih:

- *ADD ATTRIBUTE <definicija atributa>*
- *DROP ATTRIBUTE <naziv atributa>*
- *ADD <specifikacija originalne metode>*
- *ADD <specifikacija redefinisane metode>*
- *DROP <naziv metode>*

Konstruisani tipovi

- *Konstruisani tipovi u SQL:1999 standardu su referentni tipovi (reference), tipovi vrste i kolekcije.*
- *Konstruišu se pomoću konstruktora tipova REF, ROW i ARRAY.*
- *Konstruisani tipovi mogu biti atomski i složeni (referentni tip je atomski tip, a tip vrste i kolekcija su složeni tipovi).*
- *U postojećoj verziji SQL standarda podržana je samo jedna vrsta kolekcije i to niz.*

Konstruisani tipovi

- *Referentni tip*
- Efekat identiteta objekata ostvaren je u SQL:1999 standardu uvođenjem koncepta referentnog tipa, odnosno reference.
- Tabele definisane direktno nad struktuiranim tipovima mogu imati *referentnu kolonu*, koja služi kao identifikator n-torki. Referentna kolona može biti primarni ključ tabele ili, na primer, kolona sa jedinstvenim vrednostima koje automatski generiše SUBP.
- Da bi se podržalo referenciranje n-torki tabela sa referentnim kolonama omogućeno je da atributi budu definisani nad tipovima podataka koji su referentni tipovi.

Konstruisani tipovi

- *Referentni tip*
- Referenciranju može biti određen opseg (SCOPE), koji se specificira navodjenjem naziva relacije čije se n-torke referenciraju:

A REF(T) SCOPE R

Referentna kolona tabele čije se n-torke referenciraju određuje se dodavanjem sledeće klauzule u CREATE TABLE naredbu tabele:

REF IS <naziv atributa> <nacin generisanja>

Konstruisani tipovi

- *Referentni tip*

Naziv atributa je naziv dat koloni koja će služiti kao "identifikator objekta" za kolonu. Način generisanja je tipično:

- · SYSTEM GENERATED, sa značenjem da je SUBP odgovoran za održavanje jedinstvenosti vrednosti kolone u svakoj n-torki
- · DERIVED, sa značenjem da će SUBP koristiti vrednosti primarnog ključa relacije za izvodjenje jedinstvenih vrednosti kolone

Konstruisani tipovi

- *Tip vrsta*
- niz polja koja čine parovi (<naziv podatka>, <tip podatka>)
- novina u SQL:1999 je to da je sada moguće definisati promenljive i parametre koji su tipa vrsta, odnosno definisati kolonu u tabeli koja će imati kompleksnu strukturu

ROW (<naziv polja, tip polja> [{, <naziv polja, tip polja>} ...])

Tip vrste u tabeli može:

- predstavljati vrstu tabele ili
- pripadati jednoj koloni tabele kao složeni tip podatka

Konstruisani tipovi

- *Tip vrsta*

Primer:

```
CREATE TABLE adresa (ulica CHAR(30),  
                        broj INTEGER,  
                        grad CHAR(20));
```

```
CREATE TABLE student (br_indeksa CHAR(6),  
                        ime CHAR(15),  
                        prezime CHAR(15),  
                        adresa ROW (ulica CHAR(30),  
                                    broj INTEGER,  
                                    grad CHAR(20)));
```


Konstruisani tipovi

- *Kolekcija*
- Kolekcija je grupa koja se sastoji od nula ili više elemenata istog tipa.
- Broj elemenata kolekcije se naziva kardinalnost kolekcije
- Niz A je uredjena kolekcija u kojoj je svaki element povezan sa tačno jednom rednom pozicijom (koja se naziva indeks)
- Dva niza su uporediva ako i samo ako su tipovi njihovih elemenata uporedivi
- Formalna definicija niza je:

<tip elemenata niza> ARRAY [<maksimalna kardinalnost niza>]

Konstruisani tipovi

- *Kolekcija*

Sledećom naredbom se ubacuje jedna n-torka u tabelu sekcija i zatim se prikazuje naziv sekcije i član sekcije na drugoj poziciji u nizu kojim su predstavljeni članovi.

```
CREATE TABLE sekcija(naziv CHAR(15),  
                      clan CHAR(20) ARRAY[20]);
```

```
INSERT INTO sekcija (naziv,clan)  
VALUES ('dramska', ARRAY ['Markovic', 'Popovic', 'Denic']);
```

```
SELECT naziv, clan[2] AS ime FROM sekcija;
```