Korišćenje predikata

Posebnu pažnju zaslužuje klauzula *WHERE*, ne samo zbog toga što može da bude uključena u sve DML naredbe, već i zato što uključuje predikate. Kao što je već rečeno, klauzula *WHERE* omogućava specificiranje uslova pretraživanja koji filtrira vrste koje ne želimo da se pojave u rezultujućoj tabeli. SQL podržava brojne tipove predikata koji testiraju da li je uslov pretraživanja tačan, pogrešan ili nepoznat.

Poređenje SQL podataka

Predikati poređenja su vrsta predikata koja poredi vrednosti u specificiranoj koloni sa datom vrednošću. Operator poređenja se koristi za poređenje ovih vrednosti. Klauzula *WHERE* može da sadrži jedan ili više operatora poređenja. Svaki predikat (bilo operator poređenja ili neki drugi) se procenjuje na individualnoj osnovi u cilju određivanja da li udovoljava uslovu koji je definisan predikatom. Posle provere predikata klauzula *WHERE* se procenjuje u celosti. Klauzula mora da se proceni kao tačna da bi vrsta bila uključena u rezultat pretraživanja, da bi bila ažurirana ili izbrisana. Ako je klauzula procenjena kao netačna ili nepoznata vrsta neće biti uključena u rezultat, odnosno neće biti promenjena ili izbrisana.

Operator BETWEEN

Operator *BETWEEN* izvršava Boolean test vrednosti prema opsegu (rangu) vrednosti. On vraća TRUE kada je vrednost uključena u opseg i FALSE kada vrednost pada izvan opsega. Rezultati su NULL (nepoznati) ako je bilo koja vrednost opsega NULL. Koristi se u sprezi sa ključnom reči *AND*.

Naredba prema sintaksi standarda SQL2003:

SELECT ...

WHERE *expression* [NOT] BETWEEN *lower_range* AND *upper_range*

Ključne reči:

WHERE *expression*

Poredi skalarni izraz, kao što je kolona, da rangira vrednosti koje su omeđane sa *upper_range* i *lower_range*.

[NOT] BETWEEN *lower_range* AND *upper_range*

Poredi izraz (*expression*) sa sa donjom granicom (*lower_range*) i gornjom granicom (*upper_range*. Poređenje je uključivo, što znači da su donja i gornja granica uključene u rezultat. Na primer, donja granica <= izraz <= gornja granica.

Operator *BETWEEN* se koristi da testira izraz na opseg vrednosti. Operator *BETWEEN* može da se koristi sa bilo kojim tipom podataka izuzev BLOB, CLOB, NCLOB, REF i ARRAY.

U ovom primeru su prikazani podaci o prijavama iz predmeta sa šifrom 5 i sa ocenom između 8 i 10:

```
SELECT * FROM Prijave WHERE Ocena BETWEEN 8 AND 10 AND Spred = 5;
```

```
Spred  Indeks Upisan Snast  Datump                 Ocena
------ ------ ------ ------ ----------------------- ------
5      1      2000   4      2002-02-03 00:00:00.000 8
5      1      2002   4      2004-06-03 00:00:00.000 8
5      2      2001   4      2003-06-03 00:00:00.000 10
5      2      2002   4      2004-06-03 00:00:00.000 8
5      3      2001   4      2003-06-03 00:00:00.000 9
5      3      2002   4      2004-06-03 00:00:00.000 8
5      4      2000   4      2002-06-03 00:00:00.000 10
```

Prikazati sve podatke o studentima koji nisu upisani od 2000 do 2002:

```
SELECT * FROM Studenti WHERE Upisan NOT BETWEEN 2000 AND 2002
```

```
Indeks Upisan Imes                 Mesto                          Datr                    Ssmer
------ ------ ------------------------ ------------------------------ ----------------------- ------
5      2003   Mira                     Kragujevac                     1984-01-23 00:00:00.000 4
8      2003   Jovan                    Aranđelovac                    1984-03-21 00:00:00.000 4
16     2003   Tanja                    Kragujevac                     1984-10-07 00:00:00.000 1
17     2003   Zoran                    Kraljevo                       1984-03-14 00:00:00.000 2
18     2003   Saša                     Jagodina                       1984-07-16 00:00:00.000 3
```

Prikazati sve ocene od 7 do 10 (zaključno) za šifru predmeta 5 i datumom polaganja u periodu od prvog januara 2003. godine do 31 decembra iste godine:

```
SELECT * FROM Prijave WHERE Ocena >= 8 AND Ocena <= 10 AND Spred = 5 AND
(Datump BETWEEN '01-JAN-2003' AND '31-DEC-2003');
```

Vraćanje NULL vrednosti

Za kolone koje dozvoljavaju NOT NULL vrednosti, često je potrebno da se prikažu null vrednosti u upitima. Iz ovih razloga SQL predviđa *NULL* predikat, koji omogućava da se definiše uslov pretraživanja koji vraća null vrednosti.
Prikazati sve podatke o studentima čije mesto stanovanja nije poznato:

```
SELECT * FROM Studenti WHERE Mesto IS NULL
```

```
Indeks Upisan Imes                 Mesto                          Datr                    Ssmer
------ ------ ------------------------ ------------------------------ ----------------------- ------

(0 row(s) affected)
```

Vraćanje sličnih vrednosti

Predikat *LIKE* obezbeđuje fleksibilno okruženje u kome mogu da se specificiraju vrednosti koje su samo slične vrednostima, koje su memorisane u bazi podataka. Nije potrebno da znamo celu vrednost nekog podatka, koju želimo da dobijemo iz baze podataka. Predikat *LIKE* koristi dva simbola: % i _, koji mogu da se koriste na bilo kom mestu u vrednosti i

mogu da se kombinuju. Način na koji se koriste ova dva simbola određuje tip podatka, koji se selektuje iz baze podataka.

Operator LIKE

Operator *LIKE* omogućava specifičnim nizovnim šablonima u naredbama *SELECT, INSERT, UPDATE* i *DELETE* da se uparuju, posebno u klauzuli *WHERE*. Specificirani šablon može da uključi specijalne džoker znakove (simbole). Specifični džokeri su različito podržani od platforme do platforme.

Sintaksa prema standardu SQL2003

WHERE *expression* [NOT] LIKE *string_pattern*

[ESCAPE *escape_sequence*]

Ključne reči

WHERE *expression* LIKE

Vraća Boolean TRUE kada se vrednost izraza (*expression*) slaže sa šablonom znakova (*string_pattern*). Izraz može da bude kolona, konstanta, promenljiva jezika domaćina, skalana funkcija ili konkatenacija bilo čeka od ovoga. To ne bi trebalo da bude korisnički definisani tip, niti bi trebalo da budu određeni tipovi LOB_a.

NOT

Čini suprotnim predviđanje. Naredba vraća Boolean TRUE ako vrednost izraza ne sadrži šablon znakova, a FALSE ako vrednost izraza sadrži šablon znakova.

ESCAPE *escape_sequence*

Omogućava pretraživanje za postojanje znakova koji bi normalno bili interpretirani kao džokeri.

Uparivanje šablona znakova je lako sa *LIKE*, ali postoji nekoliko jednostavnih pravila kojih se treba setiti:

- Svi znakovi, uključujući zadnje i vodeće blenkove (*spaces*), su važni.
- Ne slaganje tipova podataka može da se poredi korišćenjem *LIKE*, ali oni memorišu znakovne šablone različito. Posebno, treba biti na oprezu sa razlikama između tipova podataka CHAR, VARCHAR i DATE.
- Korišćenje *LIKE* može da negira indekse ili prinudi DBMS da koristi alternativne, manje optimalne indekse nego prave operatore poređenja.

ANSI standard aktuelno podržava dva džokera operatora koji su podržani od većine platformi:

% označava nula ili više nepoznatih znakova,
_ *(underscore)* donja crta predstavlja samo jedan nepoznat znak.

Prikazati sve podatke o nastavnicima, koji bilo gde u svom imenu imaju 'ko'

```
SELECT * FROM Nastavnici WHERE Imen LIKE '%ko%'
```

```
Snast  Imen
------ -------------------------
4      Marković
5      Nikolić
7      Janković
```

Pozivanje proširenih izvora podataka

SQL podržava nekoliko tipova predikata koji omogućavaju pozivanje izvora na drugačiji način od onih u glavnoj tabeli, koje želimo da vidimo ili menjamo. Kao rezultat može da se kreira uslov pretraživanja, koji poredi podatke između tabela na način da odredi, koje vrste će biti uključene u rezultat upita, koje vrste će biti ažurirane i koje će biti obrisane. U ovu grupu predikata spadaju *IN* i *EXIST*.

Operator IN

Operator IN omogućava da se odredi da li su vrednosti u specificiranoj koloni jedne tabele sadržani u definisanoj listi, ili su sadržani u okviru druge tabele. Operator *IN* obezbeđuje način za skiciranje liste vrednosti, bilo navedenih eksplicitno ili iz podupita i poređenje vrednosti nad listom u klauzuli *WHERE* ili *HAVING*.

Sintaksa naredbe prema standardu SQL2003

{WHERE | HAVING | {AND | OR} }

value [NOT] IN ({*comp_value1*, *comp_value2* [,...] | *subquery* })

Ključne reči

{WHERE | HAVING | {AND | OR} } *value*

Dopuštena bilo pod *WHERE* ili *HAVING* klauzulom. Poređenje *IN* može, takođe, da bude deo klauzule *AND* ili *OR* u više uslovnoj klauzuli *WHERE* ili *HAVING*. *Value* može da bude bilo kog tipa podataka, ali je obično ime kolone tabele koja je pozvana transakcijom, ili moguće promenljive jezika domaćina kada se koristi programski.

NOT

Opciono, govori bazi podataka da pogleda da li podaci, u listi klauzula *WHERE* ili *HAVING*, nisu sadržani u listi vrednosti ili rezultatu pod upita.

IN ({*comp_value1*, *comp_value2* [,...] | *subquery* })

Definiše listu kompratívnih vrednosti (otud *comp_value*) za poređenje nad njima. Svaka *comp_value* mora da bude isto ili kompatibilno tipu podataka kao u početnoj (inicijalnoj) *value*. One su takođe upravljane pravilima standardnih tipova podataka. Na primer, vrednosti nizova moraju da budu ograničene sa navodnicima, dok celobrojne vrednosti ne moraju. Kao alternativa za navođenje određenih vrednosti, moraju se koristiti zagrade da zatvore podupit koji vraća jednu ili više vrednosti kompatibilnog tipa podataka.

Prikazati sve podatke o predmetima koji imaju šifre 1, 2, 6, 7, 14:

```
SELECT * FROM Predmeti WHERE Spred IN (1,2,6,7,14)
```

```
SPRED  NAZIVP
------ -------------------------
1      Logika
2      Algebra
6      Diskretna matematika
7      Numerička analiza
14     Diferencijalna geometrija
```

Operator EXISTS

Operator *EXIST* testira podupit za postojanje vrsta. i pored velike sličnosti postoji mala razlika u odnosu na predikat *IN*. Operator *EXIST* je usmeren samo na određivanje da li pod upit vraća vrstu(e) ili ne. Ako vraća sračunava se u *TRUE*, inače u *FALSE*. U okviru vrednosti koje se

Sintaksa operatora prema standardu SQL2003:

SELECT ...

WHERE [NOT] EXISTS (*subquery*)

Parametri i ključne reči su kako sledi:

WHERE [NOT] EXISTS

Testira podupit za postojanje jedne ili više vrsta. Ako čak jedna vrsta zadovoljava uslov podupita, vraća se vrednost Boolean TRUE. Opciona ključna reč NOT vraća vrednost Boolean TRUE kada podupit vraća ne uparene vrste.

subquery

Vraća rezultujući skup koji se bazira na potpuno formiranom podupitu.

Operator *EXISTS* proverava podupit na postojanje jedne ili više vrsta prema vrstama u upitu tabele roditelj.

Prikazati sve podatke o nastavnicima koji nisu nigde angažovani:

Pošto su u instanci tabele Nastavnici svi nastavnici angažovani, upisaće se u bazu novi nastavnik, a potom izvršila naredba sa predikatom *EXISTS*:

```
USE Studije
INSERT INTO Nastavnici VALUES (20, 'Stefanović')
SELECT * FROM Nastavnici n WHERE NOT EXISTS (SELECt snast FROM Angazovanje
a WHERE n.Snast = a.Snast)
```

```
Snast  Imen
------ -------------------------
20     Stefanović
```

Primer pokazuje da operator EXISTS može da sadrži više tabela. Naravno, može da sadrži i podupite.

Zapazimo da zvezdica (džoker) u podupitu je prihvatljiva, pošto podupit treba samo da vrati jedan rekord koji daje vrednost Boolean TRUE. Kolone su irelevantne (nevažne) u ovom slučaju. Prvi primer selektuje samo jednu kolonu. Ključno pitanje je da li vrsta postoji.

EXISTS, u mnogim upitima, čini istu stvar kao *ANY*. *EXISTS* je obično efektniji sa korelacionim podupitima.

Napomena: *EXISTS* je semantički ekvivalentan sa operatorom *ANY*.

Kvantifikovani operatori poređenja (*quantified comparison predicate***)**

Kvantifikovani operatori poređenja, zajedno sa operatorima poređenja, proveravaju da li bilo koja, ili sve vrednosti, zadovoljavaju uslov pretraživanja. SQL podržava sledeće operatore ovog tipa: *SOME*, *ANY* i *ALL*. Ovi operatori ne podržavaju inverzne uslove kao ostali predikati, ali može da se koristi operator (<>) što će dati isti rezultat.

Operator *ALL* izvršava Boolean test podupita za postojanje vrednosti u svim vrstama. Operator *ANY* i sinonim *SOME* izvršavaju Boolean test podupita za postojanje vrednosti u bilo kojoj testiranoj vrsti.

Sintaksa naredbe po standardu SQL2003:

SELECT ...

WHERE *expression comparison* {ALL | ANY | SOME} (*subquery*)

Ključne reči:

WHERE *expression* testira skalarni izraz, kao što je kolona, prema svakoj vrednosti u podupitu za *ALL*, i prema svakoj vrednosti dok se ne nađe slaganje za *ANY* i *SOME*. Sve vrste moraju da se spare sa izrazom da bi se vratila Boolean TRUE za *ALL* operator, dok jedna ili više vrsta se uparuju sa izrazom za *ANY* i *SOME*, da bi bila vraćena vrednost Boolean TRUE.

Comparison

> Poredi izraz u podupitu. *Comparison* mora da bude standardni operator poređenja kao što su =, <>, !=, >, >=, <, ili <=.

Operator *ALL* vraća vrednsot Boolean TRUE kada se dogodi jedna od dve stvari: bilo koji podupit vraća prazni skup (to jest nema vrsta), ili svaka vrsta u skupu udovoljava poređenje. *ALL* vraća FALSE kada se bilo koja vrsta u skupu ne slaže sa vrednošću poređenja. Operatori *ANY* i *SOME* vraćaju Boolean TRUE kada se najmanje jedna vrsta u podupitu slaže sa operacijom poređenja i FALSE kada nema vrsta, koje se slažu sa operacijom poređenja (ili kada podupit vraća prazan rezultujući skup).