

PRIVREMENE TABLICE U POHRANJENIM ZADACIMA

Privremena tablica je vrsta tablice koja se sastoji od definiranog broja atributa te može sadržavati određeni skup n-torki. To nije tablica koja je trajno pohranjena u bazi podataka već je vezana uz sesiju u kojoj je definirana. Rad s privremenim tablicama (čitanje, pisanje, ažuriranje, brisanje) identičan je radu s tablicama pohranjenim u bazi. Obzirom da je privremena tablica vezana uz sesiju odnosno korisnika koji je sesiju inicirao, moguće je da više korisnika istovremeno koristi različite privremene tablice istog imena. Korisnik iz druge sesije ne može pristupati privrmenenoj tablici iz prve sesije. Zatvaranjem sesije u kojoj je privremena tablica napravljena, tablica će biti obrisana.

DDL naredbe za stvaranje odnosno brisanje privremene tablice iste su kao naredbe za rad 'običnim' tablicama, osim što u sebi sadrže ključnu riječ TEMPORARY.

Izrada tablice

Osnovna sintaksa:

```
CREATE TEMPORARY TABLE ime privremene tablice (ime atributa tip podatka, ...)
```

Ako želimo da privremena tablica ima iste atribute kao neka tablica iz baze podataka, moguće je i sljedeće:

```
CREATE TEMPORARY TABLE ime privremene tablice LIKE ime originalne tablice
```

Brisanje tablice

```
DROP TEMPORARY TABLE [IF EXISTS] ime privremene tablice
```

Primjer:

Potrebno je napisati proceduru koja ispisuje sve nazive kvarova u zadanom odjelu. Odjel je potrebno zadati preko njegovog naziva. Osim naziva kvara potrebno je ispisati na koliko se naloga odnosi kvar te opasku 'često', 'povremeno' ili 'nikad'. Ako se kvar odnosi na više od 20 naloga potrebno je ispisati 'često', ako se odnosi na manje od 20 naloga potrebno je ispisati 'povremeno' te ako se ne odnosi niti na jedan nalog, potrebno je ispisati 'nikad'.

```
DELIMITER %%

DROP PROCEDURE IF EXISTS klasificirajKvar %%

CREATE PROCEDURE klasificirajKvar (IN zadaniOdjel VARCHAR(50))

BEGIN

DECLARE kraj INT DEFAULT 0;

DECLARE sif, br INT DEFAULT NULL;

DECLARE naziv VARCHAR(50) DEFAULT NULL;

DECLARE k CURSOR FOR SELECT sifKvar, nazivKvar FROM kvar

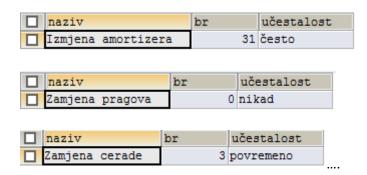
WHERE sifOdjel=
```

```
(SELECT sifOdjel FROM odjel WHERE nazivOdjel=zadaniOdjel);
                          /*obzirom da je odjel zadan preko njegovog naziva,
                            koristeći podupit saznajemo šifru odjela te definiramo
                            kursor samo za kvarove sa tom šifrom odjela*/
        DECLARE CONTINUE HANDLER FOR NOT FOUND SET kraj=1;
        OPEN k:
        vrti:LOOP
                 FETCH k INTO sif, naziv;
                 IF kraj=1 THEN LEAVE vrti;
                 END IF;
                 SELECT COUNT(*) INTO br FROM nalog WHERE sifkvar=sif;
                 IF br >= 20
                          THEN SELECT naziv, br, 'često';
                 ELSEIF (br>0 AND br<20)
                         THEN SELECT naziv, br, 'povremeno';
                 ELSE
                         SELECT naziv, br, 'nikad';
                 END IF;
        END LOOP:
        CLOSE k;
END; %%
DELIMITER ;
```

Primjer poziva procedure za odjel Mehanika:

```
CALL klasificirajKvar ('Mehanika');
```

Rezultat poziva procedure:



Poziv procedure rezultirat će sa 10 skupova rezultata jer je toliko različitih kvarova evidentirano u tablici kvar za odjel Mehanika. Razlog leži u tome što se prilikom svake iteracije petlje poziva SELECT naredba koja daje jedan skup rezultata (SELECT naziv, br, 'često' ili SELECT naziv, br, 'povremeno' ili SELECT naziv, br, 'nikad').

Ovakav je rezultat nepregledan i nepraktičan te se javlja potreba za optimizacijom izgleda izlaznih podataka. Ideja je izbjeći ispisivanje podataka naredbom SELECT unutar svake iteracije petlje, privremeno ih pamtiti, te ih ispisati tek na kraju procedure ili po završetku petlje. Ulogu privremenog spremnika podataka preuzet će privremena tablica.

Umjesto ispisa rezultata SELECT naredbom u svakoj iteraciji, taj ćemo rezultat u obliku n-torke unijeti u privremenu tablicu koju je potrebno prethodno stvoriti sa potrebnim brojem kolona. Tek po izlasku iz petlje, rezultat ispisujemo na način da ispišemo sve n-torke iz privremene tablice.

```
DELIMITER %%
DROP PROCEDURE IF EXISTS klasificirajKvar %%
CREATE PROCEDURE klasificiraj Kvar (IN zadani Odjel VARCHAR (50))
BEGIN
        DECLARE kraj INT DEFAULT 0;
        DECLARE sif, br INT DEFAULT NULL;
        DECLARE naziv VARCHAR(50) DEFAULT NULL;
        DECLARE k CURSOR FOR SELECT sifkvar, nazivKvar FROM kvar
                         WHERE sifOdjel=
                         (SELECT sifOdjel FROM odjel WHERE nazivOdjel=zadaniOdjel);
        DECLARE CONTINUE HANDLER FOR NOT FOUND SET kraj=1;
        DROP TEMPORARY TABLE IF EXISTS klasificiraniKvarovi;
        /*ako već postoji privremena tablica u sesiji imena klasificiraniKvarovi,
         prije CREATE naredbe ćemo ju obrisati*/
        CREATE TEMPORARY TABLE klasificiraniKvarovi
                 (nazKvar VARCHAR(50), brNaloga INT, učestalost VARCHAR(20));
        OPEN k;
        vrti:LOOP
                 FETCH k INTO sif, naziv;
                 IF kraj=1 THEN LEAVE vrti;
                 END IF;
                 SELECT COUNT(*) INTO br FROM nalog WHERE sifKvar=sif;
                 IF br>=20 THEN
                         INSERT INTO klasificiraniKvarovi VALUES
                                  (naziv, br, 'često');
                 ELSEIF (br>0 AND br<20) THEN
                         INSERT INTO klasificiraniKvarovi VALUES
                                  (naziv, br, 'povremeno');
                 ELSE INSERT INTO klasificiraniKvarovi VALUES (naziv, br, 'nikad');
                 END IF;
        END LOOP;
        CLOSE k;
        SELECT * FROM klasificiraniKvarovi;
END; %%
DELIMITER ;
```

Primjer poziva procedure za odjel Mehanika:

CALL klasificirajKvar ('Mehanika');

nazKvar	brNaloga	učestalost
Izmjena amortizera	31	često
Zamjena pragova	0	nikad
Zamjena cerade	3	povremeno
Zamjena prednjeg stakla	0	nikad
Zamjena krova	26	često
Promjena osovine	9	povremeno
Zamjena vozačkih i suvozačkih vrata	7	povremeno
Zamjena vozačkih sjedala	34	često
Zamjena blatobrana	18	povremeno
Učvršćivanje hvata prikolice	4	povremeno



Pokazano rješenje je i metoda kako iz funkcije koja inače vraća jedan rezultat možemo (prividno) vratiti više rezultata. Funkciji će se svakako definirati njen standardni izlaz ali osim toga, pohranjivat ćemo podatke u privremenoj tablici koje ćemo naknadno ispisati.

Primjer:

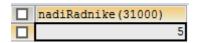
Potrebno je napisati funkciju koja će ispisati koliko radnika stanuje u zadanom mjestu. Također je potrebno ispisati i popis tih radnika. Mjesto je potrebno zadati preko poštanskog broja.

```
DELIMITER %%
DROP FUNCTION IF EXISTS nadiRadnike %%
CREATE FUNCTION nadiRadnike (ulaz INT) RETURNS INT
DETERMINISTIC
BEGIN
        DECLARE kraj, br1, br2 INT DEFAULT 0;
        DECLARE sif INT DEFAULT NULL;
        DECLARE ime, prez VARCHAR(50) DEFAULT NULL;
        DECLARE k CURSOR FOR SELECT sifRadnik, imeRadnik, prezimeRadnik
                            FROM radnik WHERE pbrStan=ulaz;
        DECLARE CONTINUE HANDLER FOR NOT FOUND SET kraj=1;
        DROP TEMPORARY TABLE IF EXISTS radniciIzMjesta;
        CREATE TEMPORARY TABLE radniciIzMjesta
                 (sif INT, imeR VARCHAR(50), prezimeR VARCHAR(50));
        OPEN k;
        SELECT FOUND ROWS () INTO br1;
        vrti:LOOP
                FETCH k INTO sif, ime, prez;
                IF kraj=1 THEN LEAVE vrti;
                END IF;
                 INSERT INTO radniciIzMjesta VALUES (sif, ime, prez);
        END LOOP;
        CLOSE k;
        RETURN (SELECT COUNT(*) FROM radniciIzMjesta);
END; %%
DELIMITER ;
```

Primjer poziva funkcije za mjesto s poštanskim brojem 31000:

```
SELECT nadiRadnike(31000);
```

Rezultat pozivanja funkcije:





Poziv funkcije rezultirat će ispisom broja radnika koji stanuju u zadanom mjestu. Popis svih radnika nalazi se u privremenoj tablici radnicilzMjesta, čiji ćemo sadržaj ispisati SELECT naredbom izvan tijela funkcije:

SELECT * FROM radniciIzMjesta;

Rezultat ispisa svih podataka iz privmrene tablice *radnicilzMjesta*:

sif	imeR	prezimeR
305	Nikolina	Horvat
358	Aljoša	Marijanić
399	Tino	Zerec
452	Viktor	Kuzmić
493	Vlado	Burela