

KONTROLA TOKA POHRANJENIH ZADATAKA

1. Kontrola toka

Kontrolom toka moguće je preusjeravati tok izvođenja pohranjenog zadatka u ovisnosti je li dani uvjet ispunjen. MySQL unutar pohranjenih zadataka omogućava kontrolu toka kroz IF i CASE uvjetovanje.

IF uvjetovanje

Osnovan oblik kontrole toka pohranjenog zadatka jest pomoću IF uvjetovanja. U slučaju da je uvjet ispunjen izvršava se jedan skup naredbi, a u slučaju da nije, izvršava se drugi skup definiran pomoću ELSE izjave ili se ulazi u novo grananje pomoću ELSEIF izjave.

Osnovna sintaksa:

```
IF uvjet
    THEN naredbe
    [ELSEIF uvjet1 THEN naredbe]
    ...
    [ELSE naredbe]
END IF;
```

Primjer (IF uvjetovanje s jednim grananjem):

Potrebno je napisati funkciju koja će dohvatiti trenutno vrijeme s poslužitelja, te ako je dan u tjednu (ponedjeljak do petak), ispisati odgovarajuću poruku.

```
DELIMITER //
CREATE FUNCTION danUTjednu()
RETURNS VARCHAR(255) DETERMINISTIC
BEGIN
    DECLARE poruka VARCHAR(255);
    IF DAYOFWEEK(NOW()) BETWEEN 2 AND 6 THEN
        SET poruka = 'Danas je radni dan';
    END IF;
    RETURN poruka;
END //
DELIMITER ;
```

Primjer poziva punkcije:

```
SELECT danUTjednu();
```

Rezultat poziva funkcije:

<input type="checkbox"/>	danUTjednu()
<input type="checkbox"/>	Danas je radni dan

Primjer (IF uvjetovanje s više grananja):

Potrebno je napisati funkciju koja će dohvatiti trenutno vrijeme s poslužitelja, te ispisati odgovarajuću poruku o tome koji je trenutno dan u tjednu.

```
DELIMITER //
DROP FUNCTION IF EXISTS danUTjednu //
CREATE FUNCTION danUTjednu()
RETURNS VARCHAR(255) DETERMINISTIC
BEGIN
    DECLARE poruka VARCHAR(255);
    IF DAYOFWEEK(NOW()) = 2 THEN
        SET poruka = 'Danas je ponedjeljak';
    ELSEIF DAYOFWEEK(NOW()) = 3 THEN
        SET poruka = 'Danas je utorak';
    ELSEIF DAYOFWEEK(NOW()) = 4 THEN
        SET poruka = 'Danas je srijeda';
    ELSEIF DAYOFWEEK(NOW()) = 5 THEN
        SET poruka = 'Danas je četvrtak';
    ELSEIF DAYOFWEEK(NOW()) = 6 THEN
        SET poruka = 'Danas je petak';
    ELSEIF DAYOFWEEK(NOW()) = 7 THEN
        SET poruka = 'Danas je dubota';
    ELSE
        SET poruka = 'Danas je nedjelja';
    END IF;
    RETURN poruka;
END //
DELIMITER ;
```

Primjer poziva punkcije:

```
SELECT danUTjednu();
```

Rezultat poziva funkcije:

<input type="checkbox"/>	danUTjednu()
<input type="checkbox"/>	Danas je ponedjeljak

CASE uvjetovanje

U slučajevima kada postoji potreba prilikom uvjetovanja kosititi više grananja, umjesto IF uvjetovanja praktičnije će biti koristiti CASE uvjetovanje.

Osnovna sintaksa (1):

```
CASE
    WHEN expression_1 THEN commands_1
    ...
    WHEN expression_n THEN commands_n
    ELSE commands
END CASE
```

Osnovna sintaksa (2):

```
CASE expr
    WHEN val_1 THEN commands_1
    ...
    WHEN val_n-1 THEN commands_n-1
    ELSE val_n
END CASE
```

Primjer:

Potrebno je napisati funkciju koja će dohvatiti trenutno vrijeme s poslužitelja, te ispisati odgovarajuću poruku o tome koji je trenutno dan u tjednu.

```
DELIMITER //
CREATE FUNCTION danUTjednu() RETURNS VARCHAR(50)
DETERMINISTIC
BEGIN
    DECLARE vrati VARCHAR(50);
    CASE DAYOFWEEK(CURDATE())
    WHEN 2 THEN SET vrati='Danas je ponedjeljak';
    WHEN 3 THEN SET vrati='Danas je utorak';
    WHEN 4 THEN SET vrati='Danas je srijeda';
    WHEN 5 THEN SET vrati='Danas je četvrtak';
    WHEN 6 THEN SET vrati='Danas je petak';
    ELSE SET vrati='Danas je vikend';
    END CASE;
    RETURN vrati;
END; //
DELIMITER ;
```

Primjer poziva funkcije:

```
SELECT danUTjednu();
```

Rezultat poziva funkcije:

<input type="checkbox"/>	danUTjednu()
<input type="checkbox"/>	Danas je ponedjeljak

2. Petlje

Petlje u pohranjenim zadacima omogućavaju višekratno ponavljanje skupa istih naredbi, ovisno o tome je li određen uvjet ispunjen ili nije. Moguće su tri vrste petlji kao što je opisano u nastavku.

WHILE petlja

WHILE petlja iterira definirane naredbe sve dok je zadovoljen određeni uvjet, pri čemu se taj uvjet provjerava na početku iteracije petlje.

Osnovna sintaksa:

```
WHILE uvjet DO  
naredbe  
END WHILE
```

Primjer:

U primjeru je napisana procedura u kojoj se WHILE petlja iterira 5 puta. U svakoj se iteraciji povećava vrijednost varijable *var* za 1, te se ispisuje sa SELECT naredbom.

```
DELIMITER $$  
CREATE PROCEDURE proc()  
BEGIN  
    DECLARE var INT;  
    SET var=1;  
    WHILE var<=5 DO  
        SET var=var+1;  
        SELECT var;  
    END WHILE;  
END;  
$$  
DELIMITER ;
```

Primjer poziva procedure:

```
CALL proc();
```

Poziv procedure rezultirat će sa 5 skupova rezultata odnosno sa ispisom vrijednosti: 2, 3, 4, 5, 6.

REPEAT ... UNTIL petlja

REPEAT petlja iterira definirane naredbe sve dok se ne zadovolji određeni uvjet, pri čemu se taj uvjet provjerava na kraju iteracije petlje.

Osnovna sintaksa:

```
REPEAT
    naredbe;
    UNTIL uvjet
END REPEAT
```

Primjer:

U primjeru je napisana procedura u kojoj se LOOP petlja iterira 5 puta. U svakoj se iteraciji povećava vrijednost varijable *var* za 1, te se ispisuje sa SELECT naredbom.

```
DROP PROCEDURE proc;
DELIMITER $$
CREATE PROCEDURE proc()
    BEGIN
        DECLARE var INT;
        SET var=1;
        REPEAT
            SET var=var+1;
            SELECT var;
        UNTIL var>5
        END REPEAT;
    END;
$$
DELIMITER ;
```

Primjer poziva procedure:

```
CALL proc();
```

Poziv procedure rezultirat će sa 5 skupova rezultata odnosno sa ispisom vrijednosti: 2, 3, 4, 5, 6.

LOOP petlja

LOOP petlja izvršavat će niz naredbi sve dok ne dođe do prekida izazvanog LEAVE izjavom. Ako se LEAVE koristi u kombinaciji sa IF uvjetovanjem, moguće je definirati petlju koja se izvode sve dok se ne ispuni određeni uvjet. ITERATE izjavom prelazi se u sljedeću iteraciju petlje od trenutka poziva izjave ITERATE.

Osnovna sintaksa:

```
Ime_petlje: LOOP
    Naredbe;
    [IF uvjet THEN
        LEAVE ime_petlje;
```

```

        END IF;]
    [IF uvjet THEN
        ITERATE ime_petlje;
    END IF;]
END LOOP;

```

Primjer:

U primjeru je napisana procedura u kojoj se LOOP petlja iterira 5 puta. U svakoj se iteraciji povećava vrijednost varijable *var* za 1, te se ispisuje sa SELECT naredbom.

```

DROP PROCEDURE proc;
DELIMITER $$
CREATE PROCEDURE proc()
    BEGIN
        DECLARE var INT;
        SET var=1;
        petlja:LOOP
            SET var=var+1;
            SELECT var;
            IF var>5 THEN
                LEAVE petlja;
            END IF;
        END LOOP;
    END; $$
DELIMITER ;

```

Primjer poziva procedure:

```
CALL proc();
```

Poziv procedure rezultirat će sa 5 skupova rezultata odnosno sa ispisom vrijednosti: 2, 3, 4, 5, 6.