SYSTEMY BAZ DANYCH
SQL

TEMAT 22-02: Funkcje sterowania przepływem w MySQL.

Autor dokumentu: Wojciech Galiński	poniedziałek, 12 grudnia 2016 r.	351203 Technik informatyk
ŹRÓDŁA WIEDZY:	http://dev.mysql.com/doc/, http://webmaster.helion.pl/index.php/kurs-mysql.	

Zagadnienia obowiązkowe

1. **Funkcja „IF()”** – umożliwia warunkowe użycie wyrażeń języka SQL. Oto wzorzec:

   IF(*warunek, wynik_gdy_PRAWDA, wynik_gdy_FAŁSZ***)**

   Tak jak i instrukcja „**CASE**”, funkcja „**IF**” może występować także w różnych klauzulach: „**SELECT**”, „**WHERE**”, „**ORDER BY**”, co pokazują poniższe przykłady (przy czym sens używania tej instrukcji w klauzuli WHERE jest wątpliwy – można ją łatwo zastąpić zwykłym warunkiem).

   PRZYKŁADY:

```
-- Funkcja IF() w klauzuli SELECT (ręczne generowanie wartości kolumnie)
SELECT id, imie, If(Right(imie, 1)='a', 'kobieta', 'mężczyzna') AS Płeć FROM uczen;
-- Funkcja IF() w klauzuli ORDER BY (sortowanie wg parzystości w długości imienia)
SELECT id, imie FROM uczen ORDER BY If(Length(imie)%2=0, 1, 0);
```

2. **Instrukcja „CASE”** – działa, jak wielokrotnie użyta instrukcja „**IF**” (analogicznie do innych języków programowania). Można sprawdzać m. in.: wiele wartości jednego warunku (wersja I) oraz wiele warunków (wersja II).

   Oto podstawowe wzorce:

   WERSJA I:

```
CASE [wartosc_lub_wyrazenie]
  WHEN wartosc_1 THEN wynik_1
  [WHEN wartosc_2 THEN wynik_2]
   …
  [ELSE wynik_N]
END
```

   WERSJA II:

```
CASE
  WHEN wyrazenie_1 THEN wynik_1
  [WHEN wyrazenie_2 THEN wynik_2]
   …
  [ELSE wynik_N]
END
```

   Instrukcję „**CASE**” można stosować m. in. po następujących klauzulach:

   - ***po klauzuli „SELECT”*** – w tym podpunkcie pokazane są przykłady pokazujące obie odmiany instrukcji „**CASE**”;

   PRZYKŁADY:

```
-- CASE w klauzuli SELECT z warunkiem po CASE (płeć ucznia w ostatniej literze jego imienia)
SELECT  id,  imie,
        (CASE RIGHT(imie, 1) WHEN 'a' THEN 'kobieta' ELSE 'mężczyzna' END) AS Płeć
  FROM uczen;
-- CASE w klauzuli SELECT z warunkami po WHEN (przedziały wartości w osobnej kolumnie)
SELECT  id, imie,
        CASE
          WHEN id<4 THEN '1-3'  WHEN id>=4 AND id<7 THEN '4-6'  WHEN id>=7 AND id<10 THEN '7-9'
          ELSE '10-'
        END AS Przedziały
  FROM uczen
  ORDER BY imie;
```

```
-- CASE w klauzuli WHERE (rodzaje wieku)
SELECT  imie,  wiek,
        CASE
          WHEN wiek IS NULL THEN 'brak danych'
          WHEN wiek<11 THEN 'dziecko'
          WHEN wiek<20 THEN 'nastoletni'
          WHEN wiek<65 THEN 'wiek produkcyjny'
          ELSE 'emeryt/emerytka'
        END AS Przedziały
  FROM uczen
  ORDER BY wiek;
```

```
-- CASE w klauzuli WHERE (rodzaje wzrostu)
SELECT  imie,  wzrost,
        CASE
          WHEN wzrost<152 THEN 'karłowaty'
          WHEN wzrost<163 THEN 'niski'
          WHEN wzrost<174 THEN 'średni'
          WHEN wzrost<185 THEN 'wysoki'
          ELSE 'bardzo wysoki'
        END AS Przedziały
  FROM uczen
  ORDER BY wzrost;
```

- ***po klauzuli „WHERE”***;

PRZYKŁAD:

```
-- CASE w klauzuli WHERE (rodzaje wieku)
SELECT  imie,  wiek
  FROM uczen
  WHERE CASE
          WHEN wiek IS NULL THEN 'brak danych'
          WHEN wiek<11 THEN 'dziecko'
          WHEN wiek<20 THEN 'nastoletni'
          WHEN wiek<65 THEN 'wiek produkcyjny'
          ELSE 'emeryt/emerytka'
        END = 'nastoletni';
```

```
-- CASE w klauzuli WHERE
-- (wyświetlanie pojedynczego rodzaju wzrostu)
SELECT  imie,  wzrost
  FROM uczen
  WHERE
    CASE
      WHEN wzrost<152 THEN 'karłowaty'
      WHEN wzrost<163 THEN 'niski'
      WHEN wzrost<174 THEN 'średni'
      WHEN wzrost<185 THEN 'wysoki'
      ELSE 'bardzo wysoki'
    END = 'niski';
```

- ***po klauzuli „ORDER BY”***.

PRZYKŁAD:

```
-- CASE w klauzuli ORDER BY (najpierw rekordy nieparzyste, a później parzyste)
SELECT id, nazwisko, imie FROM uczen ORDER BY (CASE id%2 WHEN 0 THEN 100*id ELSE id END);
```

3. **Funkcja „IFNULL()”** – służy przede wszystkim do zastąpienia wartości „NULL” inną wartością (działanie odwrotne do funkcji „***NULLIF***”). Gdy „***wartosc_lub_wyrazenie***” przyjmuje wartość „***NULL***” zostaje zastąpione wartością „***wartosc_gdy_1_parametr_IS_NULL***”.

   Oto wzorzec:

   IFNULL(*wartosc_lub_wyrazenie, wartosc_gdy_1_parametr_IS_NULL***)**

   PRZYKŁAD:

```
SELECT imię AS Imię, IFNULL(wiek, 'brak danych') AS Wiek FROM uczen;
SELECT imie, IFNULL(stypendium, '-') AS stypendium FROM uczen;
```

4. **Funkcja „NULLIF()”** – zwraca wartość „**NULL**”, gdy obydwa parametry mają tą samą wartość. W przeciwnym wypadku zwracana jest wartość pierwszego parametru. Zatem możemy ją wykorzystać, aby zamienić określoną wartość na wartość „NULL”.

   Oto wzorzec:

   NULLIF(*wartosc_lub_wyrazenie_1, wartosc_lub_wyrazenie_2***)**

   PRZYKŁAD:

```
SELECT NULLIF(1,1), NULLIF(1,2), NULLIF(2,1);
SELECT imie, NULLIF(id_klasa, 33) AS id_klasa FROM uczen;
```

Pytania kontrolne

1. Do czego służy i jak działa instrukcja „CASE”. Które elementy tej instrukcji są wymagane, a które są opcjonalne? W którym miejscu instrukcji może występować wyrażenie warunkowe od którego zależy zwracana wartość (wynik)?
2. Do czego służy i jak działa funkcja „IF”?
3. W jakich klauzulach instrukcja „CASE” oraz funkcja „IF” są dozwolone?
4. Jakie są różnice pomiędzy instrukcją „CASE”, a funkcją „IF”.
5. Opisz budowę i działanie funkcji „IFNULL” oraz „NULLIF”. Jakie są podstawowe różnice pomiędzy tymi funkcjami?

Zadania

1. Wyświetl informacje o wysokości stypendiów uczniów lub informację o braku stypendium jako tekst: „Brak stypendium”.
2. Wyświetl imiona uczniów, zamieniając imię „Jan” na wartość „NULL”.
3. Wyświetl informacje, czy uczeń zaliczył semestr (WSKAZÓWKA: średnia ocen nie mniejsza od 1.8). Użyj kolumny „Zaliczenie” z wartościami: „tak” i „nie”.
4. Wyświetl oceny uczniów następującymi skrótami: cel – 6, 5.7; bdb – 5.3, 5, 4.7; db – 4.3, 4, 3.7, dst – 3.3, 3, 2.7; dop – 2.3, 2, 1.7; ndst – 1.3, 1, nkl – 0. Inne wartości ocen zastąp wartością „NULL”.