Dotazy v SQL

Jazyk SQL (*Structured Query Language*) je již delší dobu standardem v oblasti definování schémat databází, zadávání operací nad daty a formulování dotazů. Jazyk SQL vytvořili zaměstnanci firmy IBM v polovině 70. let. Jazyk existuje v řadě verzí.

V dalším textu bude proto podrobně popsán pouze příkaz SELECT určený pro tvorbu dotazů. Bude zde popsán příkaz SELECT ve verzi relačního databázového systému MySQL.

Syntaxe definice dotazu v jazyce SQL

Textový zápis dotazu tvoří příkaz **SELECT** z jazyka SQL:

SELECT [**DISTINCT**] seznam položek
FROM výčet tabulek
[**WHERE** podmínka]
[**GROUP BY** seznam výrazů]
[**HAVING** podmínka]
[**UNION** příkaz **SELECT** bez ORDER BY]
[**ORDER BY** popis uspořádání]

Příkaz SELECT v MySQL

```
SELECT pozadavky FROM podminky_vyberu;SELECT pozadavky FROM podminky_vyberu1 UNION SELECT pozadavky FROM podminky_vyberu2;
```

Př.: ***SELECT autor FROM knihovna;*** - tento příkaz nám vybere z tabulky "knihovna" všechny autory.

Př.: ***SELECT autor, kniha FROM knihovna;*** - tento příkaz nám vybere z tabulky "knihovna" všechny autory a knihy.

Pomocí **UNION** můžeme spojit výběr z dvou tabulek ("pozadavky" musí být shodné).

Za příkaz **SELECT** se zapisuje seznam výstupních údajů oddělených čárkami.

Za část **FROM** se zapisuje seznam tabulek oddělených čárkami.

Vyber vše..*

Př.: ***SELECT * FROM knihovna;***- hvězdička nám vybere z tabulky "knihovna" všechna data.

Výběr části dat podle podmínky

```
WHERE podminka;
```

Př.: ***SELECT * FROM knihovna WHERE poznamka='precteno';*** - vytáhne všechny informace o knihách které jsou přečtené "precteno"

Za **WHERE** se zapisují podmínky pro výběr ze všech záznamů.

Př.: ***SELECT knihovna.kniha FROM knihovna,cetba WHERE knihovna.kniha=cetba.kniha;*** - tabulku "knihovna" už známe, zde je navíc tabulka "cetba", která obsahuje informace o přečtených knihách - příklad nám vytáhne názvy knih z knihovny ("knihovna"), které máme v knihovně ("knihovna") a četli jsme je ("cetba")

Ve druhém příkladu slouží **WHERE pro spojování tabulek**. Jedná se pohled do druhé tabulky, který nám umožňuje propojení obou tabulek na základně shodného názvu knih v obou tabulkách.

Porovnávací operátory

- **=** (rovno), **<>** (nerovno), **<** (menší), **<=** (menší nebo rovno), **>** (větší), **>=** (větší nebo rovno)- **<=>** (rovno; včetně hodnot NULL), **!=** (nerovno; stejné jako <>
- **x BETWEEN x1 AND x2;**- určí zda se "x" nachází mezi hodnotami "x1" až "x2" (včetně těchto hodnot) - takto vypíšeme informace o knihách z knihovny, které vyšli mezi roky 1990 (včetně) a 2000 (včetně)
- **x NOT BETWEEN x1 AND x2;**- určí zda "x" je mimo hodnoty "x1" až "x2" (včetně těchto hodnot); je to tedy opak k operátoru BETWEEN
- **IN (kde_hledat)** - hledá hodnoty dle zadaného seznamu
- **NOT IN**- opak IN..
- **IS NULL;**- nulová hodnota
- **IS NOT NULL** - opak nulové hodnoty
- **LIKE** - upřesnění výběru- př.: ***SELECT kniha FROM knihovna WHERE autor LIKE 'Z%';*** - operátor LIKE vybere knihy jejichž autor začíná od Z- procento "%" nahrazuje libovolný počet znaků, podtržítko "_" pouze jeden znak
- **NOT LIKE**- opak k operátoru LIKE

Odstranění duplikátů

```
DISTINCT
```

Př.: ***SELECT DISTINCT poznamka FROM knihovna;*** - tento příklad nám vypíše jaké používáme poznámky, tedy P,N,U (bez DISTINCT by vypsal vše: P,N,U,P,P,N)

SELECT DISTINCT zabraňuje opakovanému výpisu vícekrát se vyskytujících hodnot.

Slučování do skupin

```
GROUP BY
```

Př.: ***SELECT poznamka, SUM(stran) AS 'celkem_stran' FROM knihovna GROUP BY poznamka;*** - sečte (příkaz SUM) počet stran u knih seskupených dle poznámek (P-přečteno, N-nepřečteno...)

Za **GROUP BY** se uvádí údaj, který slouží pro seskupování – tj. vytváření skupin pro potřeby zjišťování počtu, součtu či průměru hodnot.

Seřazení

```
ORDER BY podminka;
```

Př.: ***SELECT * FROM knihovna ORDER BY autor,kniha;*** - vybere z tabulky všechny informace a srovná je vzestupně podle jmen autorů a názvů knih

```
ORDER BY podminka DESC;
```

Za **ORDER BY** se zapisuje položka, která slouží jako klíč pro seřazení. **DESC** znamená sestupně (**ASC** vzestupně – to je implicitní hodnota).

Logické operátory

Výstupem jsou nalezené hodnoty, popřípadě pravdivostní hodnota: "1","true" (pravda) nebo "0","false" (nepravda)

- **AND, &&** logický součin
- **OR, || logický součet**
- **NOT, ! – negace.**

Aritmetické operátory

Přehled operátorů: **+** (součet), **-** (rozdíl), ***** (součin), **/** (podíl), **%** (zbytek po podílu)

Manipulace s čísly (agregační funkce)

- **AVG(nazev_sloupce)**- spočítá průměr numerických hodnot ve sloupci
- **COUNT(nazev_sloupce)** - spočítá počet hodnot ve sloupci
- **COUNT(DISTINCT nazev_sloupce)**- spočítá počet jedinečných hodnot ve sloupci
- **MAX(nazev_sloupce) ***maximum ve sloupci
- **MIN(nazev_sloupce)- minimum sloupce**
- **SUM(nazev_sloupce)**- provede součet číselných hodnot ve sloupci

Výběr ze skupin

Pokud nechceme všechny skupiny, provedeme výběr skupin zapsáním podmínky za **HAVING.** HAVING se muže vyskytnout jen tehdy, je-li zároveň užito GROUP BY a podmínka za HAVING je spojena s nějakou agregovanou hodnotou.

Příklad databáze

V následujícím příkladu je popsána tabulka „knihovna“, která je použita ve výše uvedených příkladech. Jako primární klíč je uveden název knihy.

autor VARCHAR(20)	kniha VARCHAR(20) NOT NULL PRIMARY KEY	stran SMALLINT UNSIGNED	rok YEAR(4)	poznamka SET ('neprecteno', 'precteno','pujceno') DEFAULT 'neprecteno'
Bílý Josef	Malujeme byt	129	2001	precteno
Černý Tomáš	Horníkův den	96	1985	neprecteno
Červený Jiří	Asertivita	198	1996	precteno,pujceno
Zelený Karel	Lesnictví	250	1999	precteno
Zelený Petr	Alkohol a my	203	2001	precteno
Žlutý Standa	Historie Číny	367	2003	neprecteno

Literatura: Materiál z www.gene.cz