DUM 14 téma: Příkazy pro vkládání a editaci

ze sady: 3 **tematický okruh sady:** III. Databáze
ze šablony: 7 – Kancelářský software **určeno pro:** 4. ročník
vzdělávací obor: 18-20-M/01 Informační technologie
vzdělávací oblast: odborné vzdělávání
metodický list/anotace: viz VY_32_INOVACE_07314ml.pdf

Naplnění a změny databáze

Již vytvořená databáze má praktický smysl, až když je naplněna daty. Pro správce databáze je okamžik, když začne být využívána, velmi kritickým – od té chvíle je totiž velmi obtížné s databází (zejména její strukturou) jakkoliv manipulovat a případné chyby v návrhu se řeší již velmi obtížně. I proto je (zvláště u rozsáhlých databázových systémů či přechodech mezi různými verzemi informačních systémů) klíčový zkušební provoz.

Stejně tak zásahy do již existujících („živých“) dat jsou velmi nebezpečnou operací, protože databázový systém není schopen rozlišit, zda příkaz uživatelem zadaný má opravdu provést natolik rozsáhlé změny, nebo se jedná o chybu či překlep. Opatrnost je namístě i proto, že s SŘBD standardně neexistuje možnost rollbacku (vrácení o krok zpět). Jednou provedená operace je tak nezvratná.

Vkládání dat

Relativně bezpečnou operací je vkládání nových řádků do tabulek (můžeme se setkat i s výrazem „datová věta“). Při vkládání je nutné alespoň rámcově znát strukturu tabulky (tabulek), do kterých data vkládáme. Zejména pak jejich integritní omezení (doménová a referenční).

Jazyk SQL používá pro vkládání datových vět především příkaz INSERT.

Základní syntaxe příkazu je:

INSERT INTO tabulka VALUES (hodnota, hodnota…)

kdy musíme znát fyzické pořadí sloupců v cílové tabulce a v tomto pořadí předávat serveru i jednotlivé hodnoty datové věty. Zároveň je při tomto způsobu nutné uvést hodnoty všech sloupců, i v případě že zapisujeme prázdné hodnoty.

Příklad:

INSERT INTO zamestnanci VALUES (“Pavel“,‘‘Skocdopole‘‘, 25, 26400)

kde zjevně první sloupec tabulky obsahuje jméno, dále příjmení, poté třeba věk a posledním údajem může být plat. Textové hodnoty zapisujeme do uvozovek, číselné hodnoty pak přímo.

Jinou variantu, kdy chceme vložit data pouze do některých sloupců tabulky lze použít, ovšem pozor na sloupce s vlastností „NOT NULL“, které musí být vyplněny vždy. Syntaxi

příkazu rozšíříme o seznam a pořadí sloupců, které vyplňujeme – zde lze i změnit pořadí jednotlivých vyplňovaných hodnot:

INSERT INTO tabulka (sloupec, sloupec…) VALUES (hodnota, hodnota…)

konkrétně v příkladu:

INSERT INTO zamestnanci (jmeno, prijmeni) VALUES (''Frantisek'', ''Skocdopole '')

hodnoty ostatních sloupců v řádku budou prázdné, nebo budou mít výchozí (default) hodnotu (například 0, NULL atd.). Zde se také ukazuje vhodnost pojmenovat sloupce jednotně malými písmeny, bez diakritiky a mezer.

Poslední variantou je konstrukce s direktivou SET, která se ale častěji používá při změnách existujících sloupců.

Aktualizace existujících dat v tabulce

Méně bezpečnou činností jsou pak změny již zapsaných dat v tabulce či tabulkách. Znovu je nutné připomenout, že SŘBD provádějí změny okamžitě a bez možnosti návratu k předchozímu stavu. U kritických aplikací se proto změny provádějí na kopii dat a až po jejich dokončení a kontrole (například kontrolní součet, nebo kontrole člověkem) jsou původní data nahrazena.

K aktualizacím slouží příkaz UPDATE se základní syntaxí

UPDATE *tabulka* SET *sloupec=hodnota, sloupec=hodnota…* WHERE *podmínka*

Kde je naprosto nutné ověřit přítomnost a funkčnost omezující podmínky! V případě absence klausule WHERE, nebo při její chybné konstrukci (kdy by jí vyhovovaly například všechny řádky tabulky) totiž databázový systém provede bez upozornění změny u všech řádků!

U živých a kritických databází se mnohdy před provedením UPDATE nejprve volá SELECT se stejnou podmínkou a software ověřuje, zda nemá výsledek „nečekaně mnoho“ řádků.

Obecně také není vhodné pomocí update měnit obsahy primárních klíčů a podobných strukturně-kritických hodnot. V praxi se spíše pomocí kombinace INSERT a SELECT vytvoří kopie řádku s jiným primárním klíčem, provedou se na ní případné další změny a původní věta je následně odstraněna.

Konkrétní příklad aktualizace může vypadat takto:

UPDATE zamestnanci SET plat=25400 WHERE osobni_cislo=1234

Po provedení příkazu již není způsob, jak předchozí hodnotu sloupce „plat“ zjistit, nebo změny vrátit. Pro databázovou selekci (podmínku WHERE) byl použit jednoznačný identifikátor, předpokládáme, že „osobni_cislo“ je v dané tabulce primárním klíčem.

Zásadní chybou by bylo použití nevhodné podmínky:

UPDATE zamestnanci SET plat=25400 WHERE jmeno LIKE „Tomas“

Kdy lze očekávat, že takový příkaz zasáhne hned několik zaměstnanců zároveň a logicky nemá smysl.

Použití předchozí hodnoty

Při aktualizacích v databázi lze v příkazu do určité míry využít i předchozí hodnoty datového pole – jelikož je v okamžiku přijetí příkazu ještě známa. To se týká zejména hodnot číselných. Při konstrukci takového dotazu je třeba věnovat pozornost logické posloupnosti matematických a logických operátorů. Uvedeme příklad v zjednodušeném skriptovacím jazyce:

$a = SELECT pocet FROM sklad WHERE id_vyrobku=852

$a = $a + 1

UPDATE sklad SET pocet=$a WHERE id_vyrobku=852

….

který lze ovšem zapsat obecně přímo v rámci SQL jazyka, nezávisle na původním software:

UPDATE sklad SET pocet=pocet+1 WHERE id_vyrobku=852

Úkoly pro samostatnou práci

- sestavte příkaz, který všem zaměstnancům z oddělení „uctana“ zvýší plat o 10%
- sestavte příkaz, který v tabulce se sloupci „jmeno“, „prijmeni“, „oddeleni“, „telefon“ a „login“ vytvoří nového zaměstnance Pavla Nováka s loginem „novakp“, který ještě není přiřazen do žádného oddělení a nemá ani telefon

- sestavte příkaz, který výše uvedenému panu Novákovi přidělí telefon 312248752
- sestavte příkaz, který pana Nováka přejmenuje na Vomáčku (problematiku případné změny loginu zanedbáme)
- sestavte příkaz, který pana Vomáčku přiřadí do oddělení 12
- firma se sloučila s jinou a stávající oddělení byla přejmenována - před stará čísla oddělení bylo doplněno číslo „5“; sestavte příkaz, který provede změnu pro celou firmu naráz (předpokládáme, že firma má nejvýše 99 oddělení)

Zdroje:

- Archiv autora