Oponentský posudek diplomové práce

Martin Paľo: Přínosy objektově relační technologie

Cílem práce bylo dle zadání studium a porovnání technik reprezentace informací v relačních a objektově-relačních databázových systémech, zvláště z hlediska efektivnosti práce uživatele a funkce systému. Získané poznatky měly být prezentovány na vybraných úlohách.

Tomuto zadání odpovídá. i struktura práce. Za hlavními kapitoly lze považovat kapitolu čtvrtou, obsahující popis obecných vlastností a požadavků na OR databázový systém, následující dvě kapitoly, popisující jejich realizaci především v databázi Oracle a v menší míře také v normě SQL:1999, Informixu, a DB2. Stěžejní kapitolou této práce je potom kapitola sedmá, popisující vlastní testy, souběžně realizované pomocí relační a objektově relační technologie nad OR serverem Oracle 9i a srovnání jejich efektivity.

Za referenční aplikaci byla zvolena podmnožina studijního informačního systému fakulty, realizovaná částečně na datech reálných (přednášky, semináře a cvičení), částečně na datech generovaných (osobní údaje, zapsané předměty, známky a podobně).

Autor provedl velké množství testů na dotazy různého zaměření a složitosti (od jednoduchých SELECTů až po nalezení studijního plánu studenta na daný rok s ohledem na prerekvizity, korekvizity a povinnost předmětů. Z nich v textu práce prezentuje kolem třiceti případů.

Testy jsou prováděné nad aplikacemi, navrženými jednak jako relační aplikace nad relačním modelem, jednak jako objektová aplikace nad objektově orientovaným datovým modelem. Na stranách 48 (obr. 7.1) a 49 (obr. 7.2) jsou uvedené oba modely. Podle popisu na straně 50 by relační tabulce STUDIUM měla odpovídat OR tabulka OSTUDIUM, ale tu jsem na obrázku 7.1 nenašel, stejně jako hnízděnou tabulku NT_ZKOUSKA, která by měla odpovídat relační tabulce ZKOUS.

Ačkoli je část textu práce věnována popisu dědičnosti objektů v OR databázích, navržená objektová aplikace ani reprezentované testy tento aspekt příliš nevyužívají. Soustředí se na použití strukturovaných typů s minimální dědičností, a využití referencí, kolekcí a hnízděných tabulek. Testy na složitost vývoje i doby běhu aplikace, využívající rozvětvené a hluboké hierarchie objektových typů by byla jistě přínosem.

Osobně bych uvítal ještě třetí model, vzniklý z objektově orientovaného návrhu datové vrstvy jejím přímým převodem do modelu relačního. Takový model by byl dle mého názoru odlišný – složitější – než relační model použitý v práci, a dovolil by m.j. i další srovnání složitosti návrhu objektové aplikace nad relačním modelem.

Velkosti použitých tabulek jsou v sekci 7.4.4 na straně 78. Očekával bych je spíše na konci sekce 7.2. Obě velikosti naplnění databáze se lišících asi 5x až 10x, ale nepostřehl jsem , pro kterou variantu jsou uváděna data v práci. Předpokládám, že spíše pro tu větší. Rozhodně by také bylo vhodné uvést všude časy pro obě velikosti. Zajímavé by bylo vědět i to, jak bude databáze reagovat na zvyšující se počty řádek. Když už jsem u časů, textu by prospělo, kdyby byly všechny časy uváděné v jednotném tvaru. Většinou autor používá zápis v podobě hh:mm:ss.cc, ale třeba v tabulce 7.14 na str. 82 jsou časy uváděné v nějaké bezrozměrné veličině. Podle velikosti čísel v řádech 10^9 a více bych předpokládal, že se jedná zřejmě o mikrosekundy. V tabulce 7.11 na str. 77 jsou ve výpisech časové informace s jednotkou hsecs, ale nepředpokládám, že by se mělo jednat o hektosekundy.

V testech se (nijak překvapivě, ale zato velmi výrazně) jako výhoda OR přístupu ukázala možnost tvorby doménových indexů a jejich využití pro specifické úlohy jako např. full-textové vyhledávání. Závěry, vyplývající z testu 3 (sekce 7.3.1.1, str. 60) se ale do shrnutí v poslední kapitole nedostaly.

Vhodné by bylo pojmenovat v testovací aplikaci všechna integritní omezení. V některých případech (sekce 7.3.1.2, str. 62 a dalších) v prezentovaných plánech provedení vystupují anonymní indexy se jmény SYS_C*nnnnnn*, které si čtenář bez přístupu k databázi jen těžko dekóduje do srozumitelné podoby.

Jako poměrně zajímavé se jeví testy, zaměřené na množství současně otevřených kurzorů pro řešení shodné úlohy v obou verzích aplikace. Čím je menší spotřeba kurzorů v OR verzi zapříčiněna?

Je zajímavé, že relační dotaz z testu 4 (sekce 7.3.2.1, str. 63), který spojuje tabulky ZKOUS, STUDIUM a OSOBA, pro dohledání studia osoby používá UNIQUE INDEX SCAN,
Zatímco dle mého názoru jeho jednodušší varianta v testu 7 (sekce 7.3.6.1, str. 66) spojuje pouze tabulky ZKOUS a STUDIUM, ale používá FULL SCAN na obou tabulkách a následně HASH JOIN. Jak moc byla data v obou případech shodná? Podobně je to v objektově relačních verzích, kde se v obou případech spojují stejné tabulky. Přesto se oba plány provedení také dost liší. Nezkoušel jste optimalizátor donutit, aby indexy použil? Výsledky by potom mohly být odlišné.

Celkově se domnívám, že předkládaná práce splnila kladené zadání. Autor vybral oblast zahrnující různé aspekty OR rozšíření relační databáze, s výjimkou samotné dědičnosti, které ukázaly některé zajímavé rysy implementace OR rozšíření v Oracle 9. Provedl množství testů, ale prezentace jejich výsledků by mohla být lepší a především srozumitelnější pro širší okruh čtenářů. V některých případech autor také narazil na nesnadno řešitelné problémy, zda použité dotazy jsou skutečně nejefektivnější možné a dávají odpovědi skutečně nejrychleji, respektive, zda optimalizace zvolené databáze dokázala sama o sobě vybrat skutečně nejefektivnější plán provedení. Přes své výše uvedené výhrady k prezentaci autorových výsledků se domnívám, že práci je možné uznat jako práci diplomovou.

V Praze dne 5. 9. 2006

RNDr. Michal Kopecký, Ph.D.
KSI MFF UK