Objektově relační databáze a ORACLE 8

Ludmila Kalužová

VŠB - TU Ostrava, Ekonomická fakulta, Katedra informatiky v ekonomice, Sokolská 33, 701 21 Ostrava 1

Abstrakt

V současné době existuje velký počet fungujících aplikací relačního typu. Zároveň však jsou známy i nedostatky relačního konceptu, které mají za následek hledání dokonalejší formy mapování objektivní reality. Jedno z možných řešení je objektová orientace databázových systému cestou rozšíření relační koncepce. Objektově relační přístup umožňuje verze 8 databázového systému firmy ORACLE. Prvky objektového rozšíření jsou stručně specifikovány v příspěvku.

1. ÚVOD

Objektově relační databáze jsou v literatuře řazeny k ***třetí generaci databázových systémů.*** Tato generace zahrnuje systémy, které čerpají z objektově orientovaného přístupu (druhou generací se chápe čistě relační koncepce a první pak síťová a hierarchická koncepce).

Požadavky na objektově relační databáze byly specifikovány již v roce 1990 Manifestem třetí generace databázového systému, který prezentovala skupina odborníků v databázové oblasti a dále tyto požadavky konkretizoval M. Stonebraker v roce 1996. Hlavní rysy objektové orientace databázových systémů jsou zde prezentovány spíše cestou rozšíření stávající relační koncepce, než její úplnou náhradou. Jedná se o:

- zajištění kompatibility s existujícími relačními databázemi
- ponechání jazyka SQL
- volitelně primární klíče
- existenci pohledů
- mechanizmus tvorby relací.

Dále se doporučuje vypustit z datového modelu jakékoliv fyzické charakteristiky konkrétního systému řízení báze dat, což vlastně podporuje normu ANSI SPARC z roku 1973. Konečně navrhuje převzít progresivní prvky z objektově orientovaného přístupu, kterými jsou komplexní datové typy, dědičnost objektů, zapouzdření objektů a akceptace funkcí, perzistentní programovací jazyk.

Firma Oracle Corp., která je vedoucím světovým výrobcem databázového softwaru, se svou verzí 8 (uvedenou na trh v roce 1997) orientovala na rozšíření jazyka SQL o objektově orientované prvky. V následující kapitole jsou analyzovány prvky objektového rozšíření databázového systému ORACLE 8 .

2. OBJEKTOVĚ RELAČNÍ DATABÁZOVÝ SYSTÉM ORACLE 8

Jádrem databázového systému Oracle 8 zůstává relační databáze, verze 8 tedy zachovává kontinuitu směrem k předchozím verzím. Navíc jsou implementovány některé rysy objektově orientované koncepce, což dává systému charakter objektově relačního SŘBD. K prvkům objektového rozšíření patří:

- **abstraktní datové typy**

Abstraktní datový typ je tvořen jedním nebo několika atomickými typy. V relační tabulce je možné definovat sloupec, který používá definovaný abstraktní datový typ reprezentující několik atributů. Tento datový typ lze použít v deklaraci jiných tabulek. Příkladem může být definice datového typu *adresa,* která je složena z atributů *směrovací číslo, město, ulice, číslo domu:*

```
CREATE TYPE adresa_ty AS OBJECT
        (psc VARCHAR2(5),
         mesto VARCHAR2(20),
         ulice VARCHAR2(25),
         cislo NUMBER(5));
```

Uvedený datový typ lze použít například při deklaraci tabulky osoba:

```
CREATE TABLE osoba
        (idcislo VARCHAR2(5),
         prijmeni VARCHAR2(25),
         jmeno VARCHAR2(15),
         adresa adresa_ty);
```

Systém pak vytváří metody pro řízení dat, což jsou programy pojmenované podle datového typu

Při výběru konkrétního atributu abstraktního datového typu je třeba provést jeho kvalifikaci jménem sloupce tabulky:

```
SELECT adresa.psc
FROM osoba
        WHERE jmeno = ´KOVÁŘ´;
```

Abstraktní datové typy lze použít k definování objektových tabulek. Více tabulek může sdílet stejný typ, typy mohou být hierarchicky řazeny, tedy daný typ se může odkazovat

na jiný typ. To má samozřejmě dopad na proces analýzy datových struktur, neboť abstraktní datové typy je třeba identifikovat jako takové a pak určit struktury, v nichž budou tyto typy vnořeny. K abstraktním datovým typům mohou být dále definovány příslušné metody určující chování dat vážících se k typu.

- **kolekce**

Kolekcemi se zde rozumí *proměnná pole a vnořené tabulky.* **Vnořená tabulka** je vyjádřena jako sloupec jiné (hlavní) tabulky. Ke každému řádku hlavní tabulky může existovat více řádků vnořené tabulky. Takto je možné mapovat vztah kardinality 1:N. Základem pro vnořenou tabulku je definovaný datový typ. Například vnořenou může být tabulka *položky* do hlavní tabulky *objednávka:*

```
CREATE TYPE polozka_ty AS OBJECT
        (idzbozi VARCHAR2(6),
         mnozstvi NUMBER,
         smlcena NUMBER);

CREATE TYPE polozky_nt AS TABLE OF polozka_ty;

CREATE TABLE objednavka
        (cislo NUMBER(6),
         datum DATE,
         polozky polozky_nt)
         NESTED TABLE polozky STORE AS polozky_nt_tab;
```

V hlavní tabulce *objednavka* je pak tento typ přiřazen sloupci *polozky.* Definice hlavní tabulky dále musí specifikovat jméno vnořené tabulky, do níž se příslušná data uloží (*polozky_nt_tab*). Důsledkem pro uživatele je možnost realizace výběrů bez použití operace spojení s využitím funkce THE :

```
SELECT prehled.idzbozi, prehled.mnozstvi
FROM THE(SELECT polozky
         FROM objednavka
         WHERE cislo = 111) prehled;
```

Aplikace vnořených tabulek v konkrétním datovém modelu je součástí analytické úvahy. Se vnořenou tabulkou nelze pracovat samostatně, tvoří nedílný celek s hlavní tabulkou. Jejich totální uplatnění může způsobit jednak podstatné zhoršení výkonové charakteristiky databáze, jednak zkomplexnění struktury databáze prakticky znemožňující její pružnou aktualizaci (jde o vícenásobné vnoření struktury sloužící různým aplikacím).

Proměnné pole je množinou atributů stejného datového typu. Specifikuje se jako abstraktní datový typ s jedním atributem definovaným jako proměnné pole o maximálním počtu opakování. V tabulce se uvádí tento abstraktní datový typ jako jeden

sloupec. Například v tabulce *firma* bude vložen sloupec *telefon* s datovým typem proměnné pole:

```
CREATE TYPE telefony_va AS VARRAY(7) OF VARCHAR2(10);

CREATE TABLE firma
        (ico VARCHAR2(9),
         telefony telefony_va,
         ...);
```

Takto každý řádek tabulky obsahuje ve sloupci *telefony* opakující se hodnoty údaje. Pro vkládání dat do sloupce tohoto typu systém řízení báze dat vytváří příslušné metody. Nelze však specifikovat dotaz na prvky pole pomocí SELECTu, je nutno použít jazyka PL/SQL s příkazem cyklu. Výběry dat z proměnných polí nejsou tedy triviální. Samotná firma Oracle Corp. dále upozorňuje na zhoršení výkonových charakteristik databázového systému při použití proměnných polí zejména u tabulek s velkým počtem řádků (kardinalitou). Každé proměnné pole má také určitý rozměr (maximální počet hodnot), který nemusí vyhovovat všem aplikacím využívajícím téhož pole. Pak je nutno buďto poli přiřadit nejvyšší rozměr ze všech aplikací, nebo definovat více proměnných polí. Obě možnosti mají své nevýhody, ať už v efektivnosti zpracování, nebo v opakovaném využití definovaných abstraktních datových typů.

- **velké datové objekty**

Jedná se o možnost uložení velkého datového objektu (obvykle obrazové informace, zvukové sekvence, textu či prostorového obrazce) o rozsahu do 4 GB. Objekty mohou být jednoho ze čtyř typů:

  - ◊ BLOB - binární objekt
  - ◊ CLOB - znakový objekt
  - ◊ BFILE - binární data uložená mimo databázi a zpracovatelná jen pro čtení
  - ◊ NCLOB - sloupec typu CLOB podporující vícebytovou množinu znaků.

S výjimkou objektů typu BFILE lze přistupovat i k částem těchto objektů pro výběr i aktualizaci. Manipulace s velkými datovými objekty může probíhat s využitím řady prostředků: OCI (Oracle Call Interface), API (Applications Programming Interfaces), DBMS_LOB využívající jazyka PL/SQL obohaceného o specifické funkce a procedury.

- **odkazy**

V systému platí princip, že pro každý objekt je generován jedinečný identifikátor OID. Tabulka obsahující tyto objekty se v systému nazývá *objektovou tabulkou.* Namísto tradičního řešení vztahů v relačním datovém modelu pomocí primárních a cizích klíčů je pak použito **odkazů**, které jsou typickým rysem v systému při migraci relační aplikace s objektově relačním či objektovým přístupem. Objekty, na které směřují odkazy

z jiných objektů, se nazývají *referenčními objekty.* Jsou od odkazujících se objektů fyzicky odděleny. Odkazy jsou vlastně směrníky na řádky tabulky (= objekty). Objektová tabulka se vytvoří pomocí abstraktního datového typu:

```
CREATE TYPE pracovnik_ty AS OBJECT
        (id_cis VARCHAR2(5),
         prijmeni VARCHAR2(25),
         jmeno VARCHAR2(15));

CREATE TABLE pracovnik OF pracovnik_ty;
```

Odkaz na referenční objekt probíhá pomocí typového konstruktoru REF, přičemž může z daného typu objektu směřovat i více odkazů na různé typy nebo i na stejný typ objektu (to umožňuje řešit konceptuální situaci více různých vztahů mezi dvěmi entitami). Výše uvedený příklad objektové tabulky může být doplněn následujícím způsobem:

```
CREATE TYPE katedra_ty AS OBJECT
        (cis_kat VARCHAR2(3),
         nazev VARCHAR2(25),
         vedouci REF pracovnik_ty);

CREATE TABLE katedra of katedra_ty;
```

Takto je definován odkaz směřující z objektové tabulky *katedra* do objektové tabulky *pracovnik.*

Pokud z daného objektu existuje odkaz na více řádků téže tabulky, možným řešením je vytvoření vnořené tabulky odkazů. V tabulkách mohou existovat odkazy i na neexistující objekty. Pokud dojde k vypuštění řádku z tabulky, na nějž existují odkazy z jiné tabulky, odkaz na původní OID zůstává v platnosti.

Je třeba však říct, že samotná existence a využívání primárních (kandidátních) klíčů není v protikladu s principem jedinečné identifikace objektů pomocí OID. Primární klíč obecně pro daný objekt (řádek) plní jednak funkci jednoznačné identifikace jednak funkci prostředku pro jednoznačné vyhledávání objektu (řádku) uživatelem. OID identifikátor plní první z těchto funkcí. Pokud však uživatel požaduje jednoznačnou specifikaci objektu za účelem výběru či aktualizace, musí se opřít o primární (kandidátní) klíč. Dá se tedy říci, že role primárního klíče je u objektově relačních systémů oslabena, nikoliv však zcela eliminována.

- **objektové pohledy**

Objektové pohledy dovolují použít objektové koncepty na již existující relační databázi, aniž by došlo ke změně stávajících aplikací. Jedná se o prostředky pro definování objektů na existujících relačních strukturách.

- **metody**

Metoda je tvořena blokem příkazů jazyka PL/SQL sloužícím k definování přístupu k objektu. Systém Oracle ji stanovuje jako část specifikace abstraktního datového typu. Tělo metody se definuje v rámci příkazu CREATE TYPE BODY. Uživatel může používat metodu na datovém typu, pro který je definována. Metoda může být definována buďto jako *funkce* (vrací jednu hodnotu) nebo jako *procedura* (nevrací hodnotu).

3. ZÁVĚR

Tyto nové prvky přinášejí změnu v přístupu návrhu datové základny, je třeba provést analýzu návrhu datových typů s cílem jejich použití ve více tabulkách. Rovněž k těmto typům lze specifikovat metody, které definují jejich chování. Z uvedených rysů vyplývají dvě skupiny inovací. Jedna je založena na rozšíření relačního mechanismu, druhá na objektovém návrhu s možností využití existujících relačních struktur. Praxe ukáže do jaké míry nebo ke které z variant se uživatel přikloní.

LITERATURA

[1] Stonebraker, M.: Object-Relational DBMSs, The Next Great Wave, Morgan Kaufmann Publ., Inc., San Francisco, 1996

[2] Simon, A. R.: Strategic Database Technology: Management for the Year 2000, Morgan Kaufmann Publishers, Inc., 1995

[3] Koch, G., Loney, K.: ORACLE 8, The Complete Reference, Osborne McGraw-Hill Comp., Berkeley, 1997