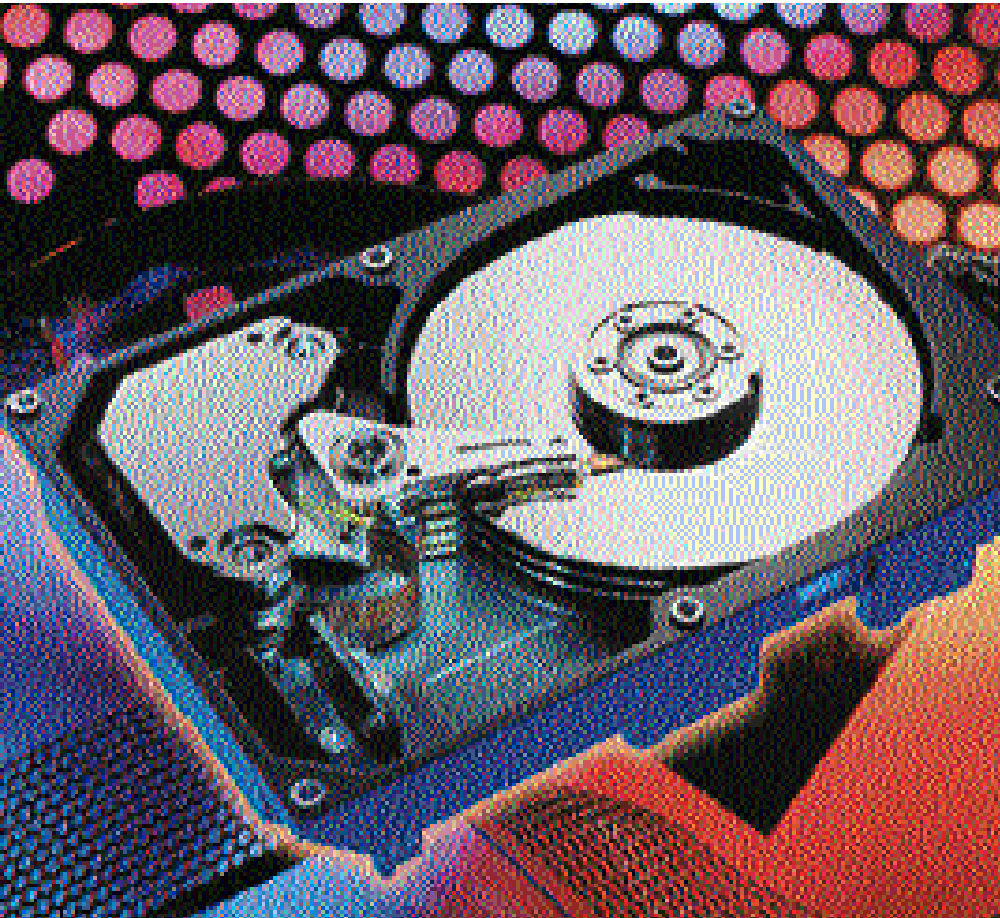


Správa rozsáhlých datových systémů

Jan Vrána



Přestože výkon počítačů prudce roste, je známo z teorie algoritmů, že stále existuje mnoho typů úloh, jejichž asymptotická složitost (vzhledem k rozsahu úlohy) roste mnohem rychleji než výkon počítačů. Proto je stále opodstatněná snaha hledat nové metody řešení některých problémů, jejichž obecné řešení už sice existuje, ale má vysokou složitost.

Složitost problému správy dat roste strmě se vzrůstajícím objemem spravovaných dat, ale roste extrémně strmě s jejich vzrůstající vnitřní složitostí, tj. se složitostí interních vazeb a vztahů. Existuje mnoho různých databázových systémů, které je možno použít jako jádro správy dat, ale zde se zaměříme na relační databázové systémy (RDBMS), hlavně protože jsou široce rozšířeny a prověřeny z hlediska výkonu a spolehlivosti.

Problém nyní zní: nalézt takovou metodu uložení logické reprezentace (modelu) spravovaného systému v RDBMS, která bude splňovat dvě základní kritéria: maximalizuje provozní výkon algoritmů správy dat,

a navíc minimalizuje náklady na vývoj a údržbu algoritmů na dané metodě založených.

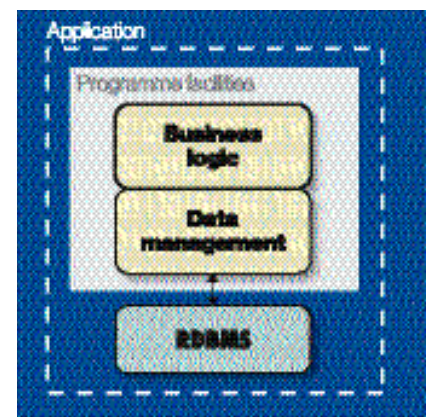
Existují dva základní (krajní) přístupy k ukládání dat s vysokou složitostí do RDBMS. Každý z nich vyniká v jednom

zmíněném kritériu, bohužel ale zároveň zcela propadá ve druhém kritériu. Většina komerčních aplikací řešících tento problém využívá nějaký přístup (bohužel nezdokumentovaný) založený na určitém kompromisu mezi těmito dvěma krajními základními přístupy. Splnit obě zmíněná kritéria není snadné a většina komerčně dostupných aplikací vykazuje špatné vlastnosti v některém z nich.

Následující odstavce popíší zmíněná dva základní přístupy ke správě rozsáhlých datových systémů a s ohledem na nejdůležitější aspekty správy dat s vysokou složitostí navrhnou novou metodu dynamického relačního uložení dat (DRD), která zachovává většinu výhod jednotlivých krajních přístupů a zároveň eliminuje většinu jejich hlavních nedostatků. Podrobný popis problematiky viz [1].

Metody ukládání a správy dat

Jak je patrné z obrázku 1, programový kód softwarové aplikace lze rozdělit na dvě části: na aplikační logiku (business logic) a na část správy dat (data management). Aplikační logika implementuje algoritmy určující chování logiky aplikace, která je



Obr. 1: Struktura aplikace

v principu nezávislá na způsobu uložení dat. Na druhé straně část správy dat implementuje algoritmy pro ukládání a manipulaci se spravovanými daty. Lze to také chápat jako transformaci mezi "objektovým" modelem

logiky aplikace a jeho technologickou reprezentací – obrazem – schopnou uložení do RDBMS).

Toto rozdělení, ale hlavně poměr mezi velikostí aplikační logiky a správy dat a také ostrost a výraznost hranice mezi oběma částmi, nám umožní lépe popsat vlastnosti jednotlivých metod uložení a správy dat. Budeme uvažovat a porovnávat následující metody:

- obecný „objektový“ přístup,
- pevnou datovou strukturu,
- dynamické relační uložení.

Dynamické relační uložení je nově navrhovanou metodou. Zbývající dvě metody budou sloužit jako referenční pro vzájemné srovnávání vlastností.

Obecný „objektový“ přístup (GOA)

Obecný „objektový“ přístup k ukládání a správě logického modelu je první z hraničních metod pro ukládání a správu dat. Zde ji zmiňujeme pro srovnání s nově navrhovanou metodou. Základní myšlenkou obecného přístupu je dosažení maximální věrnosti uloženého modelu a možnosti uložit tento model v přirozené a obecné formě, coby objekty a jejich vazby.

Všechny aspekty předmětného systému jsou zde nejprve popsány pomocí pojmů „objekt“ a „vazba objektů“ a v této formě jsou poté uloženy do relační databáze. Cílem této transformace je sjednotit, formalizovat a zjednodušit původně rozmanitý systém objektů tak, aby při zachování původní obecnosti a rozmanitosti jej bylo možné uložit do relační databáze, která vyžaduje pevnou strukturu ukládaných dat.

Touto cestou může být libovolný homogenní systém objektů a vazeb na úrovni aplikační logiky (ω -systém) transformován na jeho Ω -obraz ve formě objektů a jejich vazeb. Tato transformace je homomorfní projekcí ω -systému do jeho Ω -obrazu, takže je zaručena existence zpětné transformace. Výsledný Ω -obraz původního ω -systému může být potom snadno uložen do hostitelské relační databáze.

Metoda obecného objektového přístupu je charakteristická svou univerzálností a schopností uložit a zpracovat libovolný systém objektů s použitím poměrně jednoduché transformace. Díky jednoduchosti transformace mezi ω -systémem v aplikační logice a jeho Ω -obrazem uloženým v databázi je také relativně jednoduchá ta část aplikace, která je za realizaci transformace

zodpovědná – část správy dat. Část aplikační logiky v tomto případě dominuje nad částí správy dat a také hranice mezi oběma částmi aplikace je jasná a zřetelná právě díky obecnosti transformace.

Hlavní přednosti

Hlavní předností obecného přístupu je jeho obecnost a univerzalita. Náklady na vývoj aplikace jsou v tomto případě úměrně hlavně rozsahu a složitosti aplikační domény. Složitost a náročnost všech vývojových fází rychle klesá v pozdějších fázích života aplikace. Je také relativně jednoduché nasadit existující aplikaci do jiné aplikační domény.

Hlavní nedostatky

Špatná provozní efektivita je velmi závažným a podstatným nedostatkem aplikací založených na obecném objektovém přístupu k ukládání dat. Tento handicap podstatně omezuje použitelnost takových aplikací na aplikační domény velmi malého rozsahu. Dalšími nedostatky aplikací založených na obecném objektovém přístupu, které jsou většinou méně závažné než špatná provozní efektivita, jsou vyšší nároky na schopnost všech členů vývojového týmu abstraktně myslet, a nedostatek nástrojů pro vývoj, ladění a správu logického modelu aplikační domény, kterým je činnost aplikace řízena.

Pevná datová struktura (FDS)

FDS je druhou krajní metodou pro ukládání a správu logického modelu předmětného systému. I tuto metodu zde popisujeme pro srovnání s nově navrhovanou metodou. Hlavní myšlenkou metody FDS je dosažení maximální provozní efektivitu vytvořené aplikace.

Ukládání a správa logického modelu předmětného systému metodou FDS sestává v podstatě z vytvoření „klasického“ informačního systému „klasickým“ způsobem vytváření transakčních informačních systémů, který je dobře popsán v mnoha literárních pramenech. Při použití metody FDS je datová část předmětného systému popsána normalizovaným E-R modelem založeným na entitách a jejich relacích. Entity jsou potom uloženy jako tabulky v hostitelském RDBMS. Funkčně je předmětný systém popsán sadou pevných algoritmů, které pracují s jednotlivými vytvořenými entitami a jejich vazbami. Předmětný systém je v každé fázi považován za statický, tj. mění se pouze obsah, ale ne struktura. Díky tomu není obecně předpokládáno nasazení jednou vyvinuté aplikace (informačního systému) v jiné aplikační doméně.

Metoda FDS reprezentuje transformaci, kde je ω -systém transformován na množinu entit a relací popsanou E-R modelem (a následnou „materializací“ entit a relací v tabulkách hostitelské databáze) a množinou algoritmů, které s danými tabulkami pracují. Tato transformace je na rozdíl od obdobné transformace popsané u předchozí metody čistě statická (tj. jednorázová) a nezachovává objektový charakter ω -systému. Část správy dat na obrázku zde zabírá významný díl celé aplikace.

Metoda FDS je charakteristická vysokou specifičností vzhledem k aplikační doméně. Výsledkem je nižší pružnost, ale vyšší provozní efektivita.

Hlavní přednosti

Hlavní přednost metody FDS spočívá v možnosti dosáhnout maximální provozní efektivitu aplikace prostřednictvím efektivního využití vlastností hostitelské relační databáze a prostřednictvím možnosti individuálně optimalizovat každou operaci, což předurčuje tuto metodu k nasazení v aplikačních doménách s velkým objemem dat.

Další přednost pramení z relativně jednoduchého vývoje prvotní funkcionality aplikace. Přestože hrubá pracnost je úměrná rozsahu aplikační domény, je možné využít existující podpůrné nástroje a také zkušenosti vývojářů, zvyklých navrhovat klasické informační systémy. Vývoj také nevyžaduje schopnost abstraktního myšlení.

Hlavní nedostatky

Zásadním nedostatkem metody FDS je její strnulost a špatná adaptabilita vzhledem ke změnám aplikační domény. Na rozdíl od faktu, že vývoj prvotní funkcionality je zde relativně snadný, náklady na implementace pozdějších modifikací jsou velmi vysoké a navíc mají tendenci růst s rostoucím počtem větví aplikační logiky. Tento nedostatek má zásadní vliv na další život jednou vyvinuté aplikace.

Dynamicky relační uložení (DRD)

DRD je metoda, která z principu leží mezi oběma zmíněnými extrémy. Snaží se dosáhnout maximální obecnosti a pružnosti vzhledem ke změnám aplikační domény při současném zachování vysoké provozní efektivitu prostřednictvím maximálního využití relačních vlastností hostitelské databáze.

Správa dat podle DRD je založená na následujících principech: struktura dat aplikační domény je popsána logickým modelem

jako v případě metody GOA. Tento logický model je transformován do podoby, která je podobná technologickému modelu datové struktury aplikační domény použitým metodou FDS, tj. do systému entit a relací. Transformovaný logický model struktury dat aplikační domény je v obecné podobě uložen ve formě metadat ve zvláštní části hostitelské databáze a stává se řídicím elementem chování aplikace založené na této metodě. Jde v podstatě o formu metapopisu nebo metaprogramu [2]. Na základě uloženého metapopisu jsou potom v databázi dynamicky vytvořeny specifické datové tabulky, jejichž struktura prakticky odpovídá „pevně“ vytvořené struktuře specifických tabulek a vazeb v metodě FDS. Tato dynamicky vytvořená specifická relační struktura je potom dotazována SQL dotazy, které jsou taktéž dynamicky sestavovány podle zadaného požadavku a uloženého metapopisu. SQL dotaz sestavený tímto způsobem, který dotazuje takto vytvořenou databázovou strukturu, může plně využít vlastností hostitelské databáze.

Metoda DRD kombinuje přednosti obou krajních přístupů v tom smyslu, že data jsou uložena v „přirozené“ relační struktuře, ve které mohou být efektivně zpracovávána, přičemž je zachována možnost změnit strukturu dat i chování celé aplikace pouze změnou metapopisu. Tyto změny mohou být implementovány kdykoliv a bez jakéhokoliv zásahu do programu.

Aplikace založená na metodě DRD má podobnou strukturu jako aplikace založená na obecném přístupu. Část aplikační logiky také převažuje nad částí správy dat a hranice mezi oběma částmi je rovněž výrazná. Část správy dat je v tomto případě však poněkud větší než v případě GOA.

Hlavní přednosti

Metoda DRD spojuje nejdůležitější přednosti obou extrémních metod. Její obecnost umožňuje vytvářet aplikace, které jsou velmi obecné a z velké části nezávislé na konkrétní aplikační doméně a navíc mají velmi nízké náklady na pozdější údržbu a rozvoj. To předurčuje použití metody DRD v aplikačních doménách, které rychle a často mění strukturu. Jednou vyvinutou aplikaci lze relativně snadno nasadit do jiné aplikační domény.

Přirozeně relační způsob uložení a zpracování dat propůjčuje aplikacím založeným na metodě DRD výjimečné provozní vlastnosti a výkonnost. To předurčuje použití metody DRD v aplikačních doménách, které jsou velmi náročné na objem a strukturu dat.

Hlavní nedostatky

Hlavní nedostatek metody DRD pramení z výrazné abstraktnosti algoritmů založených na této metodě a také z nutnosti vyvinout a udržovat metadatum. Celkově vzato tyto nedostatky vyústí ve zvýšené požadavky na kvalifikovanost a schopnosti vývojového týmu.

Srovnání metod

- Metoda GOA je vhodná pouze pro aplikační domény velmi malého rozsahu s výrazně se měnící strukturou.
- Metoda FDS je vhodná pouze pro aplikační domény s relativně jednoduchou strukturou, která se nesmí měnit vůbec, nebo jen nepatrně. Aplikační doména však může mít značný objem dat.
- Metoda DRD je velmi vhodná pro vývoj aplikací určených pro správu rozsáhlých a rychle se měnících aplikačních domén. Jednou vyvinutá aplikace může být snadno a s nízkými náklady nasazena do jiné aplikační domény.

Vlastnosti všech tří metod jsou v [1] popsány na typickém případě, přičemž byly uvažovány následující výkonnostní hlediska:

- náklady na prvotní vývoj (IDC),
- náklady na údržbu a rozvoj (MDC),
- celková datová náročnost (TDC),
- metadatová režie (MDO),
- průměrná velikost metadatových tabulek (MTS),
- průměrná velikost datových tabulek (DTS),
- složitost výběru jedné hodnoty jednoho atributu (SASV),
- složitost výběru všech hodnot jednoho atributu (SAMV),
- složitost výběru aktuálních hodnot všech atributů jedné skupiny (MASV),
- složitost výběru všech platných kombinací hodnot atributů jedné skupiny (VCA),
- složitost výběru aktuálních hodnot všech atributů tabulky parametrů (AVA).

Vhodné grafické zobrazení jasně ukáže vhodné vlastnosti metody DRD [1].

Vhodné databázové systémy

Které existující databázové systémy jsou vhodné pro použití jako hostitelské databáze pro správu rozsáhlých datových systémů? Tuto otázku je možné diskutovat z mnoha různých hledisek a úhlů pohledu. Škála je velice široká. Z čistě technologického hlediska, tedy z hlediska dostatečného

výkonu a spolehlivosti, je potřeba vybrat řešení, které bude odpovídat předpokládanému rozsahu aplikační domény a předpokládané zátěže. Jedná-li se ale skutečně o rozsáhlé aplikační domény, bude zřejmě nutné sáhnout po některé z „těžkotónážních“ databází, které jsou schopny zvládnout prakticky libovolné množství dat. Nechci zde zabíhat do přílišných podrobností a srovnávání, ale určitě nebude šlápnutím vedle volba některého řešení z „velké trojky“, tedy Infomix, Oracle, DB2. Co se týče technologické platformy, velká většina velkých, a hlavně důležitých databázových aplikací je založena na některém unixovém systému, který ve velkém měřítku poskytuje dostatečný výkon, stabilitu a spolehlivost nezbytnou pro bezproblémový provoz pro organizaci často životně důležité aplikace.

Konkrétní volba je však v reálním případě spíše než technologickými parametry ovlivněna převážně parametry ekonomickými, tedy cenou pořizovací, ale také cenou provozní (cenou technické podpory), která v dlouhodobém horizontu většinou výrazně převyšuje cenu pořizovací.

Závěr

Príznivé vlastnosti metody DRD nastíněné v tomto článku nejsou jen teoretickými vývody, ale jsou podloženy reálnými pozorováními provozu reálných aplikací založených na této metodě. Za zmínku zde stojí určitě datový sklad zpracovávající statistiky vyplacených důchodů na ČSSZ a aplikace pro podporu správy konfiguračních parametrů mobilní telekomunikační sítě společnosti Eurotel jako zástupce jiného typu zpracování dat.

Literatura:

- [1] Vrána J. *Dynamic relational data storing method for management of large data systems. Doctoral dissertation. Czech University of Technology in Prague, 2003.*
 [2] Vrána J. *Methods of employing metadata for managing large systems. In DATESO'02: Proceedings of Workshop on Databases. Ostrava, 2002, p. 44–56. ISBN 80-248-0080-2.*

e-mail: jan.vrana@autori.ccb.cz
www.SystemOnLine.cz

Autor článku, Jan Vrána, je pracovníkem společnosti Komix. Článek vznikl na základě příspěvku „Effective method for management of large data systems“ publikovaném ve sborníku konference SCI 2004.