PostgreSQL a další SQL na Linuxu

Karel Žák <zakkr@zf,jcu.cz>
http://home.zf.jcu.cz/~zakkr/

SQL Databáze

Vznik SQL se datuje do roku 1970 kdy tehdejší zaměstnanec firmy IBM *E.F.Codd,* v článku *"A Relational Model of Data for Large Shared Data Banks"* publikuje teorii relačního modelu dat.

V letech 1974 - 1975 firma IBM vyvíjí prototyp relačního databázového systémému SEQUEL-IBM s dotazovacím jazykem SEQUEL (Structured English Query Language). Ze světa výzkumů do skutečné praxe se SQL dostává v roce 1979 a to v SQL systému ORACLE (od ORACLE corp.) pro počítače VAX . Přibližně v roce 1983 uvádí i firma IBM svůj databázový systém DB/2 (pro MVS).

Skutečné využívání relačních databázových systémů nastává až v polovině osmdesátých let a to díky výraznému rozvoji hardware a efektevnější implementaci SQL.

SQL databáze zvučných jmen: Oracle, Informix, Sybase, Interbase, Ingres, Progress, PostgreSQL, MySQL nebo také i Microsoft SQL Server.

Popisovat vlastní SQL jazyk je nad rámec tohoto textu. Pro lepší orientaci zde budou uvedeny jen některé základní principy.

SQL standardy

I k jazyku SQL podobně jako k jiným nástojům v oblasti výpočetní techniky byly přijaty standardy zajištující, alespoň častečnou kompatibilitu jednotlivých implementací SQL.

V roce 1986 je schválena ASCII norma. Tato norma byla v roce 1987 přijata i mezinárodní organizací ISO. Výchozí verze standardu je označována jako "SQL86" a s dodatkem z roku 1989 jako "SQL89". Dalším a novějším standardem je "SQL2" (SQL92) z roku 1992. A v současné doběje dostupná i verze SQL3.

Existence standardů a snaha o jejich vývoj je věc jistě záslužná, ale bohužel druhou věcí je úroveň podpory těchto standardů ze starany samotných SQL serverů. Není neobvyklé, že podled tvůrců SQL serverů na celou problematiku je jiný než jak uvadějí standardy nebo implementace serverů nedovoluje držet se zcela standardů. Obecně, ale lze říci, že základní SQL příkazy jsou u většiny SQL serverů stejné a naučí-li se uživatel používat jeden SQL server tak mu nebude činit větší potíže přejít na jiný.

Trochu jiná situace panuje u "administrátorských" a některých nedatových dotazů (tedy například dotazů / příkazů definujících přístupová práva k datům, způsob jejich uložení atd.). Tyto příkazy mohou buď úplně chybět (MySQL) nebo mohou být implementovány jen na základní úrovni (PostgreSQL) nebo naopak mohou být dotaženy téměř k dokonalosti (Oracle).

Trocha teorie

Relační modelový model se od síťového a hierarchického modelu liší tím, že je budovám na teoretickém základě, kterým je definovaná struktura data a operace nad těmito daty. Relační datový model je uživatelsky jednoduchým a výrazně usnadňuje manipulaci s daty, to znamená, že uživatel nemusí nijak definovat a znát algoritmus na provedení požadované operace. Také struktura dat je relativně jednoduchá a operace s daty jsou orientovány na operace s celými množinami (tabulkami) dat a nikoliv na operace s jednotlivými záznami. To je zřetelné na většině SQL příkazech, které mají jako základní parametr tabulku a nikoliv záznam, a pokud je nutné pracovat jen s určitým záznamem (nebo záznamy) tak se specifikuje podmínka která uvnitř tabulky vytvoří podmnožinu s kterou se pracuje (viz. `WHERE` u některých příkazů).

Nejjednodušší prezentací relace je tabulka. Složitější relací pak může být množina n-tic (řádků) více tabulek vybraných podle nějakého klíče.

S daty v relačních databazích se zachází na základě operací tzv. *relační algebry*, které zahrnují i klasické množinové operace:

UNION (*sjednocení*) — vzniká nová relace která obsahuje všechny řádky obou výchozích relací a řádky které jsou schodné v obou relacích jsou ve výsledné množině obsaženy jen jednou ($x \cup y$).

INTERSECTION (*průnik*) — výsledek obsahuje jen řádky společné oběma výchozím relacím ($x \cap y$).

DIFFERENCE (rozdíl) — výsledkem jsou řádky první relace, které nejsou obsaženy v relaci druhé ($x - y$).

CARTESIAN PRODUCT (*kartézský součin*) — vytvoří relaci objahující všechny řádky první relace zřetězené postupně s každým prvkem druhé tabulky[1] .

SELECTION (RESTRICTION) (*selekce*) — je výběr řádků, které vyhovují specifikované podmínce, výsledek je tedy podmnožinou původních dat.

PROJECTION (*projekce*) — výběr jen určitých atributů (sloupců).

JOIN (*spojení*) — vytvoření relace zřetězením těch řádků z jedné tabulky s takovými řádky druhé tabulky, které splňují podmínku týkající se srovnatelných sloupců v každé z tabulek dle kterých dochází k propojení.

Dle relační teorie lze pomocí základních operací (sjednocení, kartezský součin, rozdíl, selekce, projekce a spojení) uskutečnit veškeré operace s daty a ostatní operace jsou jen kombinace těchto pěti. Jak je ten svět pro ty počítače jednoduchý...

Integrita dat

Z pohledu SQL jsou rozlišovány tři druhy integrity:

- entitní integrita — zajišťuje jednoznačnou identifikaci řádku (n-tice) v tabulce (entitě) - často se jedná o tzv. `PRIMARY KEY`,
- doménová integrita — má zajistit, že data ve sloupcích budou pouze taková které jsou dle definice sloupce přípustná (např. jen určitý rozsah čísel),
- referenční integrita — je nejobtížněji implementovatelnou a udržitelnou integritu. Narozdíl od předchozích druhů integrity je tato výrazně logickou a přesahuje samotnou tabulku. Definuje vztahy mezi logicky souvisejícími entitami (tabulkami). A musí garantovat logickou konzistenci dat v tabulkách. Jedná se většinou o tzv. `FOREIGN KEY`.

V případě tvorby databáze a vztahů mezi tabulkami je integrita jednou z věcí kterou by autor databáze nikdy neměl opomenout.

Transakce

Zajišťují, že změny dat v databázi jsou považovány za platné až ve chvíli kdy jsou tyto změny uživatelem potvrzeny a nebo v případě chyby kdy je vždy obsah tabulek vrácen do stavu před započetím transakce. Toto je nutné pro udržení referenční integrity dat kdy logicky související (a závislá) data jsou vkádána ve více samostatných SQL příkazech a je tedy nutné vždy provést všechny tyto související změny najednou a nebo vůbec.

Některé dostupné SQL databáze pro Linux

Oracle (verze Oracle 8.0.5 a novější Oracle8i) — komerční databáze, která je považována z dominantní mezi všemi SQL databázemi. Jedná se o výrazně robustní SQL s imlementovanými téměř všemi vlastnostmi, které lze od SQL DB požadovat (integrita dat, transakce, stored procedury, full-text indexovaní, replikace, ...atd.). Velmi dobrá podpora i pro klientské prosředí v MS-WinXX). Je volně dostupná neomezená verze, kterou lze z licenčního hlediska provozovat 30 dní.

Sybase — také komerční, ale cenově dostupnější. Vlastnostmi podobná Oracle i když se nejedná o takového obra.

[1] Kartézský součin je typickým produktem příkazu bez specifikace vzájemného vztahu tabulek, např: `SELECT * FROM t1, t2` (a mají-li tabulky několik tisíc řádek může být vedlejším produktem též časová pauza na uvaření a vypití si kávy.)

InterBase — původně komerční databáze "střední třídy". V současné době volně dostupná a použitelná SQL, která by měla být v budoucnu open source (dostupný zdrojový kód).

PostgreSQL — plně volně šiřitelná databáze vyvíjená jako open source projekt a mající ambice zařadit se mezi robustní SQL databáze.

MySQL — volně šírítelná, ale s určitými licenčními omezeními při komerčním použití (distribuci). Starší verze jsou šířeny pod licencí GPL.

MiniSQL — volně šířítelná, malá SQL vhodná pravděpodobně je na zkoušení a nebo na velmi nenáročné projekty.

MySQL

Velmi rozšířená a často používaná SQL DB pro linux. Pro svou extrémní rychlost nejčastěji používána jako SQL pro web aplikace. Rychlost je považována za hlavní prioritu při vývoji. Bohužel dodržování SQL standardů je považováno za vedlejší a v některých i základních věcech je nekompatibilní.

Dovoluje používání všech základních SQL příkazů. Umožňuje i v celku kvalitní správu uživatelů a i má velmi dobrou podporu klientské strany (myšleno klient v klient/server). Tedy dostupné rozhraní v C, Perl, PHP, Python, ODBC ...atd.

Nemá implementované transakce a udržení integrity dat v rozsáhlejších datových modelech může být problematické. Mezi další nedostatky patří nemožnost definování vlastních funkcí a používání uživatelem vytvořenách modulů. Neumožňuje tedy použití triggerů.

PostgreSQL

Mnoho výzkumných projektů nikdy neopostí akademickou půdu. Databázovému projektu, který dnes známe pod jménem PostgreSQL se to povedlo.

Prapředkem PostgreSQL byl Ingres, který vyvíjela kalifornská univerzita v Berkeley (University of California at Berkeley) v letech 1977-1985. Tato databáze byla po čase komercializována a vznikla Ingres Corporation a jedna z prvních komerčních databází na světě.

V letech 1989-1994 se na stejné univerzitě pod vedením *prof. Michaela Stonebrakera* započal vývoj objektově-relační databáze Postgres. Tento projekt se postupně přelil do komerční podoby a je známý jako Illustra (což je tedy přímý sourozenec PostgreSQL).

Dva studenti zmíněné univerzity *Jolly Chen* a *AndrewYu* se rozhodli v letech 1994-1995 dále pracovat na akademickém kódu Postgresu a založili projekt Postgres95 (jméno pravděpodobně vzniklo od nevalné módy, která tento rok v software kruzích panovala). V létě roku 1996 přechází Postgres95 do open source podoby a je přejmenován na nezavádějící a jasné jméno PostgreSQL. Vznikají e-mailové diskuzní skupiny a PostgreSQL Global Development Team.

Od té doby je tato zcela free a open source SQL databáze vyvíjena mnoha vývojáři z celého světa. PostgreSQL z důvodu přítomnosti původního kódu z Berkeley je pod licencí této univerzity a né pod například GPL licencí. Nové verze jsou vydávány každých přibližně 3–8 mesíců. V době psaní tohoto textu byla jako poslední dostupná verze 7.0.

Jak je zřejmé tak PostgreSQL má několik bratříčků zvučného jména. Tím nejnovějším je Miriposa, databáze která je pokračovatelem Postgres95 Tato SQL se soustředí hlavně na replikace v sítích až s 10 000 SQL uzly (nejedná se, ale o realtime replkaci, ale o skládání fragmentů).

Pokud vám někdo bude tvrdit, že open source software je software, který tvoří několik nadšenců "na koleně", vězte, že ten někdo je neinformovaný nebo informovaný nechce být, protože v několika i komerčně úspěšných SQL databázích lze nalézt kód, který je v PostgreSQL a nese datum z doby, kdy někdo jiný oslavoval a učil svět pracovat z dodnes natvrdlým *command.com* zatímco již vznikal a žil multiuživatelský SQL systém, který je možné si pohodlně stáhnout z http://www.PostgreSQL.org.

PostgreSQL — vlastnosti

Robustnost — je jedna ze základních priorit vývoje. Má větší prioritu než výkon a rychlost.

SQL92/SQL3 — snahou při implementaci jazyka SQL je co nejvíce a pokud možno úplně se drže existujících standardů SQL (standard SQL2 (SQL92) a SQL3).

Modulárnost — vnitřní kód databáze (programováno v jazyce C) je výrazně modulární a rozšiřování a vývoj je tedy relativně snadný a databázovému serveru nehrozí to co je časté u některých špatně navržených projektů — tzv. “vnitřní zhroucení pod tíhou budoucího vývoje”.

Rozšiřitelnost — myšleno hlavně jako rozšiřitelnost z pohledu uživatele. PostgreSQL umožňuje uživateli definování vlastních datových typů, operátorů, funkcí a dynamické loadování modulů (podobně jako Linuxový kernel nebo apache). K psaní funkcí je možné použít tzv. procedurální jazyky (PL). Jedná se běžné programovací jazyky (s vyjímkou PL/SQL) rozšířené o několik specifických funkcí. V současné době jsou implementované PL/SQL, PL/Tcl, PL/Perl a nativně C. Pokud uživatel chce a umí to, může si vytvoři i vlastní PL jazyk. Všechna tato rozšíření je možné provádět bez re-kompilace serveru.

Velké objemy dat — PostgreSQL umožňuje vytváření tabulek a indexů bez omezení velikosti (tedy např. větší než 2Gb).

Large Objects (LO) — jedním z nepříjemných limitů PostgreSQL je omezení velikosti řádky v tabulce. Tato velikost je max. 32Kb (defaultně 8Kb). Z tohoto důvodu obsahuje PostgreSQL neobvyklé tzv. LO rozhraní. Toto rozhraní se velmi podobá standardnímu C rozhraní pro zacházení se soubory (open / close / seek). Takto uložená data pak nejsou ukládána přímo do tabulek, ale uživateli je vráceno číslo, které tento LO reprezentuje. Toto rozhraní je velmi rychlé a dovoluje ukládat i binární data bez jakýkoliv změn. Ve verzi 7.1 bude limit řádky tabulky odstraněn a libovolně velká data bude možné ukádat přímo do tabulek.

Transakce — PostgreSQL obsahuje kvalitní systém transakcí a podpory víceuživatelského přístupu k datů. Databáze má implementované zamykání na úrovni řádků (například pokud dvě nezávislé transakce provadějí `UPDATE` stejné tabulky tak jsou pro zápis zamknuty jen záznamy, kterých se update týká).

View - jedná se o zvláštní tabulku, která není reprezentována vlastní (tabulkovou) definicí a sama neobsahuje data, ale je jen náhledem nad existujícími daty v jiných tabulkách.

Rule — k dotazům typu `UPDATE, DELETE, SELECT, INSERT` je možné pro libovolnou tabulku definovat SQL operaci která se spustí na místo (nebo změní) uživatelem poležený dotaz. Toto umožňuje tzv. rewrite systém PostgreSQL.

Práva — přístupová práva je možno definovat na úrovni tabulky a to pro uživatele nebo skupiny (príkazy `GRANT/REVOKE`). Autentifikace je prováděna pomocí hesla které může být uloženi v DB, externím souboru nebo pomocí systému Kerberos.

Triggery — na libovolnou tabulku je možné “zavěsit” fuknkci, která může kontrolovat/měnit data, která jsou do tabulky vkládána.

Integrita dat — PostgreSQL od verze 7.0 implementuje všechny tři výše uvedené integrity dat, tedy i tzv. `FOREIGN KEY`.

Kursory — pro libovolný dotaz typu `SELECT` je možné definovat si kurzor, kterým je možné se ve vybraných datech pohybovat a nechat si vrátit jen požadované řádky.

Pokročilé dotazy — jedná se například o `SELECT...INTO`, `INSERT...SELECT`, `COPY`, union, subselecty, definování výsečí dotazů pomocí `LIMIT/OFFSET` a `SELECT FOR UPDATE` u příkazu `SELECT`. Dále pak například u aggregátních funkcí (max( ), count( ), ..apod) lze použí `func ( DISTINCT col )`.

Kódování — překódování dat pro kódování definované klientem. Např. databáze v LATIN2 $\leftrightarrow$ klient používá CP1250.

Date/time — rozsáhlé možnosti operací s časem (časové pásma, definování formátu, formátování (to_char()), ... apod.). Není omezení UNIXového časového rozpětí.

Přenositelnost — PostgreSQL pracuje téměř na všech dostupných UNIXových klonech a nově také na MS-WinNT.

Domumentace - PostgreSQL má velmi dobrou a rozsáhlou dokumentaci dostupnou v SGML / HTML / PS.

Klient — klientskou stranu tvoří podporovaná rozhraní v C, Perl, PHP, Python, Tcl, Java, ODBC. S PostgreSQL jsou přímo distribuované programy `psql` a pro Xwin `pgacces`, který dovoluje i určitou vizualizaci při práci s SQL dotazy.

PostgreSQL — výkon

PostgreSQL je výrazně pomalejší než např. MySQL. Je to dáno zcela jiným zpracování dotazu kdy je nutné přihlédnout k dynamickým datovým typům, operátorům, funkcím, rewrite systému a dále pak transakcím. Všechny dyto vlastnosti pochopitelne stojí určitý čas. Toto je velká odlišnost od MySQL kde je vše na pevno zakompilované a není tedy nutné psát tak robustní kód. Z těchto důvodů je tak často oblíbené porovnávání těchto SQL zcela nesmyslné. Je to porovnání jablek a hrušek. Pro porovnání je vhodnější použít některou z komerčních robustních SQL vedle kterých si PostgreSQL nevede špatně.

Vzhledem k uvedeným vlastnostem má PostgreSQL větší paměťové nároky. Za základní lze považovat instalaci na počítač s cca 128Mb RAM a PII (apod.).

PostgreSQL — instalace

Instalce je snadná. Při instalaci z package (RedHat, Debian) vše probíhá automaticky. Databáze má defaultně jeden adresář kam ukládá data (např. někde ve stromu `/var` nebo `/home`).

Při ruční instalaci je nutné definovat promněnou `$PGDATA`, která obsahuje cestu k adresáři z daty. Tento adresař se pak pod uživatelem `postgres` inicializuje skriptem `initdb`, který vytvoří základní skelet základní databáze `template1` (tato databáze pak slouží jako šablona pro každou novou DB). Pak je nutné nastartovat databázového démona a opět pod uživatelem `postgres` vytvořit prvního databázového uživatele. Tímto je možné považovat instalaci za ukončenu a je možné již DB plně ovládat pomocí SQL příkazů, např. pomocí textového klienta `psql`. Vše je velmi dobře popsáno v dokumentaci.

PostgreSQL 7.0 — co nového

- `template1` (cizí klíče)
- Pokračuje přepisování Optimalizeru (jedna z hlavních častí při zpracování dotazu).
- Zcela nový klient `psql` — textový jednoduchý klient, podpora hitorie příkazů, readline knihovny, práce z large objecty.
- Datový typ `datetime` plně nahrazen typem `timestamp` (původní `datetime` byl z důvodu kompatibility ponechán jen na úrovni parseru).
- Změny a rozšíření v JBDC a ODBC rozhraní.
- — Detailní seznam změn má několik stovek řádek.

PostgreSQL — budoucnost

- Odstranění limitů velikosti řádku. Bude ve verzi 7.1, v současné době je dostupný první zkušební patch.
- Přepsání manageru funkcí – mělo by umožnit zrychlit a zlepšit zacházení s funkcemi.
- Imlementace `OUTER JOIN`. S tím souvisí i částečné přepsání zpracování dotazu. Bude ve verzích 7.1 a 7.2.
- Systém logů (WAL), Pravděpodobě v 7.1 nebo 7.2.
- Změny v memory managementu - stabilita a úspora paměti.
- Přepsání systému ACL (access list) a umožnění "grantování" přístupových práv na úrovni sloupců. Definování práv k funkcím apod.
- Změny v storage managementu.

SQL — odkazy

- PostgreSQL: http://www.postgresql.org
- MySQL: http://www.mysql.com
- Oracle: http://www.oracle.com
- Sybase: http://www.sybase.com
- InterBase: http://www.interbase.com
- Benchmarks (nejen pro MySQL): http://www.MySQL.com/benchmark.html
- Standard SQL92: http://docs.linux.cz/sql92
- Database Jump Site (mnoho odazů na o SQL) http://www.pcslink.com/~ej/dbweb.html
- University of California, Berkeley: Database Research (Ingres, Postgres, Miraposa, Tioga, GiST ..) ftp://s2k-ftp.cs.berkeley.edu/pub/postgres/index.html