UNIVERZITA PARDUBICE

FAKULTA EKONOMICKO SPRÁVNÍ

ÚSTAV SYSTÉMOVÉHO INŽENÝRSTVÍ A INFORMATIKY

Microsoft SQL Server 2005

BAKALÁŘSKÁ PRÁCE

Autor práce: Martin Preisler

Vedoucí práce: Mgr. Karel Naiman

2007

UNIVERSITY OF PARDUBICE

FACULTY OF ECONOMICS AND ADMINISTRATION

INSTITUTE OF SYSTEM ENGINEERING AND INFORMATICS

Microsoft SQL Server 2005

BACHELOR WORK

Author: Martin Preisler

Supervisor: Mgr. Karel Naiman

2007

Prohlašuji:

Tuto práci jsem vypracoval samostatně. Veškeré literární prameny a informace, které jsem v práci využil, jsou uvedeny v seznamu použité literatury.

Byl jsem seznámen s tím, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorský zákon, zejména se skutečností, že Univerzita Pardubice má právo na uzavření licenční smlouvy o užití této práce jako školního díla podle § 60 odst. 1 autorského zákona, a s tím, že pokud dojde k užití této práce mnou nebo bude poskytnuta licence o užití jinému subjektu, je Univerzita Pardubice oprávněna ode mne požadovat přiměřený příspěvek na úhradu nákladů, které na vytvoření díla vynaložila, a to podle okolností až do jejich skutečné výše.

Souhlasím s prezenčním zpřístupněním své práce v Univerzitní knihovně Univerzity Pardubice.

V Pardubicích dne 22. 08. 2007

Preisler Martin

Anotace:

Cílem bakalářské práce je seznámení se s relačními databázemi, zejména s MS SQL Serverem 2005, zvláště pak výkonové srovnání s předchozí verzí MS SQL 2000 a otázka přechodu na novější verzi. Dále jsou uvedeny novinky v T-SQL, struktura SQL 2005 a přístup k datům v nové verzi.

Synopsis:

Ambition of this bachelor work is to get acquainted with the basis of relational databases, mainly with MS SQL Server 2005, power confrontation with previous version MS SQL 2000 and opened question of transmigration to the newer version. Further T-SQL novelties, structure of SQL 2005 and access to the database.

Obsah:

Obsah: 6
1. Úvod....8
1.1. Náhled do historie nejen relačních databází 8
1.1.1. Vznik databází 8
1.1.1.1. Model správy souborů (FMS – File Management System)....8
1.1.1.2. Hierarchický model (HDM)....8
1.1.1.3. Síťový model (SDM)....8
1.1.2. Současnost 9
1.1.2.1. Relační model RDBMS (Relational Database Management Systems) 9
1.1.2.2. Objektový model....9
1.1.2.3. Objektově-relační model....10
1.1.2.4. Post-relační model 10
1.1.2.5. Další modely 11
1.1.3. Standardizace 11
1.1.4. Normalizace 12
1.1.5. Normální formy 13
1.1.5.1. Nultá normální forma – 0. NF 13
1.1.5.2. První normální forma - 1.NF 13
1.1.5.3. Druhá normální forma - 2.NF 14
1.1.5.4. Třetí normální forma - 3. NF 15
1.1.5.5. BCNF – Boyce Coddova normální forma 16
1.1.5.6. Čtvrtá normální forma - 4. NF 16
1.1.5.7. Pátá normální forma - 5. NF 17
1.1.6. Shrnutí normalizace 17
1.2. Představení relačních databází....17
1.2.1. Výhody, nevýhody, vývoj a budoucnost 19
1.2.2. XML (eXtensible Markup Language, česky rozšiřitelný značkovací jazyk) ...20
1.2.3. Shrnutí relačních databází....21
1.2.4. Přehled relačních databází dneška 21
1.2.4.1. Oracle....21
1.2.4.2. IBM DB2 23
1.2.4.3. Sybase Adaptive Server Enterprise....24
2. Přehled jednotlivých verzí MS SQL 2000 a 2005 26
2.1. Jednotlivé verze MS SQL 2000 26
2.1.1. Verze Desktop Engine Edition 26
2.1.2. Verze Personal Edition 26
2.1.3. Verze Standard Edition....26
2.1.4. Verze Developer Edition 26
2.1.5. Verze Enterprise Edition....27
2.2. Jednotlivé verze MS SQL 2005 27
2.2.1. SQL Server 2005 Express Edition....27
2.2.2. SQL Server 2005 Workgroup Edition 27
2.2.3. SQL Server 2005 Standard Edition 27
2.2.4. SQL Server 2005 Enterprise Edition 28
2.2.5. SQL Server 2005 Developer Edition....29
2.2.6. SQL Server 2005 Compact Edition 29
2.2.7. SQL Server na platformách 64-bit....29

3. Přístup k datům ve verzi MS SQL 2005 29
3.1. ODBC (Open Database Connectivity) 30
3.2. OLE DB 31
3.3. ADO.NET 32
3.4. SQL Server Native Klient 33
4. Struktura SQL 2005 a .NET Framework 33
4.1. .NET Framework 33
4.2. Struktura SQL 2005 34
4.2.1. SQL Server Integration Services (SSIS) 34
4.2.2. SQL Server Analysis Services (SSAS) 35
4.2.3. SQL Server Reporting Services 35
4.2.4. SQL Server Notification Services (SSNS) 36
4.2.5. SQL Server Replication Services 36
4.2.6. SQL Server Database Engine 36
4.2.7. Development tools 37
4.2.8. Management tools 37
5. T-SQL a nové možnosti ve verzi MS SQL 2005 37
5.1. Co je T-SQL 37
5.2. Nejvýznamnější novinky v T-SQL 2005 38
5.2.1. Příkaz „TOP n“ 38
5.2.2. Příkazy RANK, DENSE RANK, ROW_NUMBER a NTILE 39
5.2.3. Příkaz pro ošetření chyb TRY CATCH 40
5.2.4. Triggers pro DML 41
5.2.5. Partitioning 42
5.2.6. Nový datový typ XML 42
6. Srovnávací výkonový test Microsoft SQL server verze 2000 a 2005 45
6.1. Základní údaje pro testování 45
6.2. Test 46
6.2.1. Test AS3AP 46
6.2.2. Test TPC-C 48
6.3. Testovaní redaktorů časopisu Connect 49
6.3.1. Test AS3AP 49
6.3.2. Test TPC-C 50
6.4. Závěr testování 51
7. SQL Server 2008 52
8. Závěr 53
9. Seznam tabulek 54
10. Seznam obrázků 54
11. Seznam grafů 54
12. Literatura 55

1. Úvod

1.1. Náhled do historie nejen relačních databází

1.1.1. Vznik databází

Hovoříme-li dnes o databázích, většinou máme na mysli zdroj dat – je to databáze založená na relačním modelu, popřípadě na objektovém modelu.

Historie ovšem zná i jiné možnosti ukládání dat. Jako všechny programové prostředky i databáze prošly svým vývojem. Počátky sahají do poloviny minulého století. Od té doby vzniklo několik databázových modelů, se kterými se postupně seznámíme:

1.1.1.1. Model správy souborů (FMS – File Management System)

Počátky FMS se datují do začátku druhé poloviny dvacátého století. Tento model je založen na sekvenčním ukládání dat do jednoho velkého souboru. Největší předností této varianty je jednoduchost ukládání a snadná pochopitelnost. Nevýhodou modelu správy souborů je prakticky vše ostatní, od nutnosti znát fyzické umístění jednotlivých dat, bez kterého nelze jakkoliv data vyhledat a následně s nimi pracovat, přes velkou časovou náročnost vyhledávání v databázi, po neexistenci jakékoliv kontroly referenční integrity.

1.1.1.2. Hierarchický model (HDM)

Model, vhodný pro zpracování dat, která jsou uspořádána hierarchicky. Základem tohoto modelu je uložení dat ve stromové architektuře, jednotlivé prvky jsou kořen, uzly a listy (v hierarchii jsou postaveny nejníže, nemají již další větvení). Mimo nejdůležitějšího vztahu rodič-syn existuje také vztah v rámci jedné úrovně. Tento model má oproti výše uvedenému výhodu ve snadnějším vyhledávání pomocí posloupnosti jednotlivých kroků skrz strom. Vztahy jsou charakterizovány jako 1:N jednosměrně, to znamená, že každý potomek má pouze jednoho rodiče. Výhodou oproti předešlému modelu je také odklon od nutnosti znát fyzické umístění dat. Největší nevýhodou tohoto modelu poznáme při jakékoliv změně ve stromové architektuře. Vymazání nebo změna jednoho uzlu má následky pro uzly a listy které navazují na tento uzel (kořen).

1.1.1.3. Síťový model (SDM)

Počátky tohoto modelu se datují do poloviny 60. let dvacátého století a první integrace do začátku následujícího desetiletí. Vztahy mezi jednotlivými položkami

mohou být jak 1:N tak N:M, to znamená, že vztahy mohou být jak lineární tak cyklické. Vyhledávání v tomto modelu může začít prakticky na jakémkoliv místě. Za největší nevýhodu stejně jako u hierarchického modelu můžeme považovat nesnadné provádění změn nebo mazání jednotlivých uzlů, které ovšem nemusí mít stejně fatální důsledky jako ve výše uvedeném modelu.

1.1.2. Současnost

Výše uvedené modely jsou seřazeny dle doby vzniku. Starší varianty se až na výjimky dnes již nevyužívají, ale je dobré se s nimi seznámit. Ukázaly nám některé slepé uličky a přinesly mnoho teoretických znalostí a praktických zkušeností pro své následovníky.

Dnes pracujeme se třemi základními typy databází. Jsou to databáze relační, objektové a databáze objektově-relační.

1.1.2.1. Relační model RDBMS (Relational Database Management Systems)

Dnes nejrozšířenější a nejpoužívanější datový model, kterého počátky spadají do konce šedesátých let, respektive do roku 1969, kdy doktor E. F. Codd[4)] seznámil veřejnost se svojí představou návrhu databáze založeném na matematickém aparátu relačních množin. Následně tento model implementovala IBM a byl standardizován jak dle ANSI tak dle ISO.

1.1.2.2. Objektový model

Objektové databáze jsou označované jako ODBMS (Object Oriented DBMS). Dnes je většina aplikací vytvořena v objektově orientovaných jazycích. Objekty mnohem lépe popisují komplexní realitu, urychlují vývoj aplikace a zjednodušují ji díky polymorfismu a dědičnosti. Čím komplikovanější je popis modelové skutečnosti, tím častěji se obvykle ukáže, že ji lze ztvárnit pouze objektově, nebo že E-R model (entity-relationship), tj. převedení komplexních struktur modelované skutečnosti do dvourozměrných tabulek a nalezení vztahů mezi nimi, by byl tak složitý, až by se stal v praxi nepoužitelný. Na rozdíl od E-R modelu však nemáme ucelenou teorii, která by definovala, jak objektový model navrhnout, i když už existují v praxi používané podpůrné nástroje.

1.1.2.3. Objektově-relační model

Tento model je označovaný ORDBMS "Object Relational" Database Management Systém.

Dnes se začínají oba tyto základní typy databází - objektová a relační – kombinovat. Snažíme se nalézt a přijmout z obou skupin ty lepší vlastnosti. "Rozšířená relační" či "objektově-relační" jsou synonyma pro tyto databázové systémy. ORDBMS je specifikována v rozšíření SQL standardu — SQL3. Do této kategorie patří např. IBM, Informix, Oracle či Unisys[19)].

1.1.2.4. Post-relační model

Vedle těchto základních a také nejznámějších typů databází existuje celá řada těch, které řeší práci s daty specifickým způsobem. Jádrem post-relační databáze je vícerozměrný transakční datový stroj. Ten umožňuje ukládat a velmi účinně pracovat s daty, přičemž lze zároveň používat jazyk SQL. Tento typ databázových systémů je obvykle optimalizován pro velmi rychlý vývoj a poté nasazení velmi výkonných webových aplikací typu klient-server. Příkladem může být databáze Caché. Podle výrobce[11)] srdcem platformy Caché je multidimenzionální architektura umožňující různé post-relační pohledy na spravovaná data. K datům lze díky této architektuře přistupovat nejen klasickým relačním způsobem, ale například také objektově.

Caché nabízí také skriptovací technologii CSP (Caché Server Pages). Ta je svými možnostmi srovnatelná například s ASP či JSP. Je k dispozici pro celou řadu operačních systémů, především pro Unix a Windows a její jednouživatelská verze je k dispozici zdarma. Povědomí o Caché se postupně zvyšuje také v České republice. Například Česká spořitelna používá největší řešení co se týče počtu licencí - má více než 1400 uživatelů.

Model Caché je takto nazván proto, že jednak připomíná autorům vyspělé techniky využívání vyrovnávacích pamětí – caching, což zvyšuje výkon databázového systému. Potom je to také prestiž – cachet, kterou vývojáři získají používáním této v současnosti nejlepší databázové technologie.

Databáze Caché je datovým úložištěm, kde můžeme při splnění určitých podmínek přistupovat ke stejným datům relačně i objektově. Caché 2007 obsahuje kromě jiného nový softwarový modul Jalapeño[20)], který umožňuje přidat

klasickým javovským objektům persistenci, neboli trvalost, aniž by bylo nutné provádět objektově-relační mapování[5)].

Jiným zástupcem může být databáze Oracle[10)] která rovněž umožňuje jak relační tak objektový přístup k datům s jejich výhodami i nedostatky.

1.1.2.5. Další modely

Data je také možné ukládat do souborů se specifickou strukturou – například XML či HTML, které respektují požadavky na koncepci a strukturu webu. Většina formátů je standardizována organizací w3c[12)].

Všechny uvedené místa pro uložení dat se nazývají datové sklady nebo také datová úložiště. Tato množina datových úložišť nebude zřejmě nikdy konečná, neboť neustále některé zanikají a jiné se vytvářejí.

1.1.3. Standardizace

Obecně je standardizace stanovení norem, zavádění symbolů a kódů ke zjednodušení a překonání potíží způsobených rozdílností. Pro klasické relační databáze nám stačí, když bude standardizován dotazovací jazyk. Ten zahrnuje jak jazyk pro definici, tak i pro manipulaci s daty.

Dotazovací jazyk SQL (Structured Query Language - strukturovaný dotazovací jazyk) byl ještě v roce 1989 považován jen za „zajímavé cvičení z teorie množin“, aby se během několika let stal zcela zásadním a nenahraditelným standardem v oblasti zpracování dat.

Základem jazyka SQL jsou dotazy. Odpovědí databázového systému je množina dat, která dotazu odpovídají. Velice příjemný je fakt, že dotazy v jazyce SQL mají syntaxi podobnou přirozenému lidskému jazyku.

Například:

SELECT jmeno FROM zamestnanci
WHERE vek < 45 AND plat > 25000

Tímto dotazem SŘBD žádáme server, aby nám vybral (SELECT) jména lidí z (FROM) tabulky zaměstnanců, jenž (WHERE) jsou mladší čtyřiceti pěti let (vek < 45) a jejich plat je vyšší než dvacet pět tisíc Kč (AND plat > 25000).

Síla a velikost jazyka SQL je dána hlavně tím, že jej všechny významné softwarové firmy jako standard skutečně přijaly a všechny je bez rozdílu ve svých implementacích používají. Historie jazyka začíná roku 1974. Firma IBM tehdy hledala jazyk, který by byl vhodný pro práci s databázemi a tabulkami. Vymysleli SEQUEL (Structured

English Query Language), implementován byl prototyp nazývaný SEQUEL-XRM roku 1974-1975. Tento jazyk se velice rychle rozvíjel a roku 1976-1977 vzniká nová verze (SEQUEL/2), která pak mění jméno na SQL.

Tento prototyp byl využíván pouze firmou IBM a několika vybranými klienty. Díky jeho velké oblibě, i když ještě nebyl oficiální, mnoho firem včetně Microsoftu začalo vyvíjet databáze na jeho základě. První SQL databází na trhu byl v roce 1980 Oracle pro počítače VAX. V roce 1983 firma IBM vstoupila na trh s DB2. Současně další společnosti, např. Oracle či Sybase, začalo také prodávat výrobky na základě SQL a od té doby se stal z tohoto jazyka standard databází.

Roku 1986 ANSI adoptoval SQL jako standard pro databázové jazyky a roku 1987 se stal i standardem ISO. Tato verze standardu je vedena pod jménem SQL/86. V roce 1989 byl publikován ISO (International Organization for Standartizition - mezinárodní organizace pro vývoj a sjednocení standardů) dodatek Integrity Enhancement Feature (SQL89). Další úprava proběhla v roce 1992 a začínáme hovořit o standardu SQL2, nebo také o SQL92. V roce 1999 byl tento jazyk dále rozšířen. Šlo zejména o některé objektové vlastnosti, o triggery, regulární výrazy, rekurzivní dotazy, triggery, neskalární typy. To byly verze SQL3 nebo také SQL 1999.

O verzi 2003 hovoříme od roku 2003, kdy byla představena další verze a sice s XML rozšířením. Byly také provedeny standardy v oblasti sekvencí a sloupců s automaticky generovanými hodnotami. Poslední veze z minulého roku se nazývá SQL 2006.

ANSI je nezisková organizace, která vytváří průmyslové standardy ve Spojených státech. Je členem organizace ISO a IEC. Ten fakt, že existuje standard pro databáze, nám otevírá mnoho nových možností - můžeme naše aplikace propojovat díky vysoké kompatibilitě mezi sebou i s aplikacemi cizími.

1.1.4. Normalizace

Data zachycená v primární tabulce při základním návrhu běžně obsahují informace, které se opakují a tím zabírají místo a komplikují a zpomalují vkládání, mazání a aktualizaci dat. Může dokonce dojít i ke ztrátám informací. Řešením bude rozložení (normalizace) tabulky do dvou nebo více jednodušších tabulek - relací.

Normalizace je sada pravidel, jak bychom měli postupovat při transformaci struktury modelu na strukturu fyzického uspořádání tabulek a relací v databázi. Normalizace je odstranění redundantních (opakujících) se dat, omezení složitosti (rozložením složité relace na dvojrozměrné tabulky) a zabránění tzv. aktualizačním anomáliím (např.

abychom v knihovní databázi smazáním všech knih autora nepřišli o data o autorovi). Mělo by to vést k vyšší výkonnosti, přehlednosti a rozšiřitelnosti. Také ke vzniku tabulek, které lze snadno udržovat a efektivně se na ně dotazovat. Normalizací směřujeme k tomu, že všechny závislosti původních schémat musí zůstat zachované, v relaci zůstanou zachována původní data.

1.1.5. Normální formy

1.1.5.1. Nultá normální forma – 0. NF

Definice nulté normální formy zní: "Tabulka je v nulté normální formě právě tehdy, když existuje alespoň jedno pole, které obsahuje více než jednu hodnotu."

Rasa	Výška v kohoutku	Vlastnosti
Retriever	60 cm	Společenský, hravý, nehlídá
Doga	80 cm	Společenská, přátelská, někdy hlídá

tabulka 1 - Nultá normální forma

1.1.5.2. První normální forma - 1.NF

Relace je v první normální formě, jestliže obsahuje každý její atribut jen atomické hodnoty, tedy hodnoty které jsou dále nedělitelné.

Například v následující tabulce si uvedeme porušení tohoto pravidla. Máme zde jednotlivé sklady a výrobky, které jsou v nich k dispozici:

SkladID	Adresa	PSČ	Obsah skladu
1	Pálená	531 13	Cihla, Beton, Hřebík
2	Hluboká	465 12	Beton, Hřebík, Dveře
3	Daleká 2	118 19	Hřebík, Cihla, Žebřík

tabulka 2 - První normální forma – zadání

V takovéto tabulce by se dost špatně prováděly aktualizace, případně by se špatně vyhledávalo podle skladových položek. Aby tabulka byla v 1.NF, musíme buď rozdělit atribut *Obsah skladu* do více atributů, nebo praktičtěji oddělit *Obsah skladu* a vytvořit novou tabulku. Druhé řešení je podle mne lepší, protože je podstatně flexibilnější:

SkladID	Adresa	PSČ
1	Pálená	531 13
2	Hluboká	465 12
3	Daleká 2	118 19

tabulka 3 - První normální forma - výsledná tabulka 1/2

SkladID	Obsah skladu
1	Cihla
1	Beton
1	Hřebík
2	Beton
2	Hřebík
2	Dveře
3	Hřebík
3	Cihla
3	Žebřík

tabulka 4 - První normální forma - výsledná tabulka 2/2

1.1.5.3. Druhá normální forma - 2.NF

Relace se nachází v 2.NF formě právě tehdy, jeli v první normální formě a současně každý neklíčový atribut je plně závislý na primárním klíči. A to pouze na celém klíči, nejen na jeho jakékoliv podmnožině.

Například: tabulka obsahuje následující atributy: název automobilu, jméno a rodné číslo vlastníka:

Značka_Auta	Jméno	RČ
BMW	Kouba	7914043670
Opel	Kouba	7914043670
Mazda	Kouba	7914043670
Mazda	Marek	7818013636
Opel	Dlouhý	6819083625

tabulka 5 - Druhá normální forma – zadání

Jako primární klíč by zde mohla sloužit kombinace dvou sloupců a to jméno vlastníka a jména vozu. Jak ovšem vidíme z tabulky dochází zde k porušení 2.NF, kdy třetí sloupec rodné číslo totiž nezávisí na celém klíči ale pouze na jeho části *Jméno*. To by mohlo zapříčinit aktualizační anomálii – v případě smazání aut pana Kouby, ztratíme také jeho RČ. Řešením je opět rozložení na dvě tabulky:

RČ	Jméno	AUTO_ID
7914043670	Kouba	1
7914043670	Kouba	2
7914043670	Kouba	3
7818013636	Marek	3
6819083625	Dlouhý	2

tabulka 6 - Druhá normální forma – výsledná tabulka 1/2

Auto_ID	Značka_Auta
1	BMW
2	Opel
3	Mazda

tabulka 7 - Druhá normální forma – výsledná tabulka 2/2

1.1.5.4. Třetí normální forma - 3. NF

Tabulka splňuje 3.NF právě tehdy, když splňuje dvě předešlé normy a žádný z atributů není tranzitivně závislý na primárním klíči. Mohli bychom také říci, že tabulka splňuje dvě předešlé normy a všechny neklíčové atributy jsou navzájem nezávislé.

Tranzitivní závislost je závislost mezi alespoň dvěma atributy a klíčem, kdy první atribut je funkčně závislý na klíči a druhý atribut je závislý na prvním. Například: máme informace o firemních zákaznících, relace je Zákazníci s atributy ID (primární klíč), Jméno, Zaměření, Město, PSČ, Pozice a Sleva:

ID	Jméno	Zaměření	Město	PSČ	Pozice	Sleva
11	Ondstroj	Trubky	LA	32145	Výrobce	5%
22	Kopr	Kolena	NY	32185	Velkoobchod	15%
33	Randit	Redukce	Pha	69888	Velkoobchod	15%
44	Hexan	Šroubení	UO	35487	Maloobchod	7%
55	Debit	T-kusy	NY	32185	Soukromník	3%
66	Zumr	Kolena	UO	35487	Maloobchod	7%

tabulka 8 - Třetí normální forma – zadání

V této tabulce nalezneme závislost všech atributů na primárním klíči, ale i další závislosti. Přenesená závislost atributu PSČ na klíči: ID => Město => PSČ. Další závislost atributu Sleva na klíči: ID => Pozice => Sleva. Jinak řečeno: závislosti Město => PSČ a Pozice => Sleva jsou ty, které porušují pravidlo, že všechny neklíčové atributy jsou navzájem nezávislé. Problém vyřeší opět rozklad na více relací, v tomto případě na tři, protože jsme 3. NF porušili hned dvakrát.

ID	Jméno	Změření	Město_ID	Sleva_ID
11	Ondstroj	Trubky	1	1
22	Kopr	Kolena	2	2
33	Randit	Redukce	3	2
44	Hexan	Šroubení	4	3
55	Debit	T-kusy	2	4
66	Zumr	Kolena	4	3

tabulka 9 - Třetí normální forma – výsledná tabulka 1/3

Město_ID	Město	PSČ
1	LA	32145
2	NY	32185
3	Pha	69888
4	UO	35487

tabulka 10 - Třetí normální forma – výsledná tabulka 2/3

Funkce_ID	Pozice	Sleva
1	Výrobce	5%
2	Velkoobchod	15%
3	Maloobchod	7%
4	Soukromník	3%

tabulka 11 - Třetí normální forma – výsledná tabulka 3/3

1.1.5.5. BCNF – Boyce Coddova normální forma

Boyce/Coddova normální forma se pokládá za variantu třetí normální formy. Relace se nachází v BCNF právě tehdy, jestliže pro každou netriviální závislost X=>Y platí, že X je nadmnožinou nějakého klíče schématu S.

Kandidátní klíč je ten, který je určen jedním nebo více atributy, které jednoznačně identifikují relaci. BCNF bude porušena, jestliže budou splněny tyto podmínky:

- Daná relace musí mít víc než jeden kandidátní klíč.
- V relaci musí být minimálně 2 kandidátní klíče složené z více atributů.
- Některé složené kandidátní klíče musí mít společný atribut.

Podle této definice platí, že mezi kandidátními klíči nesmí existovat žádná funkční závislost. Například můžeme mít relaci Výrobky na skladě. Tato relace má dva kandidátní klíče: {CislоOdberatele, CislоVyrobku} a {JmenoOdberatele, ČislоVyrobku}. Mezi položkami { CislоOdberatele } => {JmenoOdberatele} je funkční závislost, a to je porušení Boyce/Coddovy normální formy. Řešením je opět rozpad na více relací a to konkrétně na relace Odberatel a Vyrobek.

Tak složitých případů, kde by se něco takového mohlo stát, bude velmi málo, neboť bude model nejspíš rozložen už dříve.

1.1.5.6. Čtvrtá normální forma - 4. NF

Tabulka je ve čtvrté normální formě právě tehdy, jestliže je v BCNF a popisuje pouze příčinnou souvislost (jeden fakt). To znamená, že všechny vícehodnotové závislosti jsou zároveň funkčními závislostmi z kandidátních klíčů (v jedné relaci se nesmí spojovat nezávislé opakované skupiny).

1.1.5.7. Pátá normální forma - 5. NF

Relace je v páté normální formě právě tehdy, je-li ve čtvrté formě a nelze do ní již přidat další atribut (sloupec), nebo jejich skupinu, aniž by se vlivem skrytých závislostí rozpadla na několik dílčích relací.

1.1.6. Shrnutí normalizace

Čím více máme dat ke zpracování, tím složitější je databáze a tím více je potřeba normalizovat. Například u příkladu 3. NF by menší firma s několika desítkami zaměstnanců asi nepotřebovala dávat PSČ do další tabulky, protože by to bylo zbytečné. Výsledkem by také bylo rozdrobení databáze do spousty relací, z nichž asi velká část by byla zbytečná. Pak se s takovou složitou strukturou databáze velmi špatně pracuje. Ale v tabulce dodavatelů u firmy s jejich velkým množstvím to už význam určitě má.

V každém případě by mělo platit, že každá databáze by měla být určitě normalizovaná do 3. úrovně. U vyšších normálních forem je už člověk obvykle tak zkušený, že problém rozloží již ve fázi konceptuálního modelu a navrhne již normalizované relace.

1.2. Představení relačních databází

Relační databáze jsou takové, které vyhovující relačnímu modelu dat (RMD). Relační model dat je nejrozšířenější způsob, jak uložit data v databázi.

V tabulkách je vždy popis nějaké části reálného světa. A stejně jako ve skutečném životě, i data zde zachycovaná mohou být spolu v nějakém vztahu. Například učitelé učí několik tříd, ve třídách jsou žáci. Učitelé jsou tedy v určitém vztahu ke třídám i k jednotlivým žákům.

Jak na tyto skutečné vztahy, tak na tabulky dat a na vztahy mezi nimi (implementované opět jako tabulky) se dá pohlížet jako na matematický pojem relace. Pojem relace je založený na matematické teorii množin a predikátové logice a definuje způsob práce s daty. Jde o reprezentaci dat, způsob jejich ochrany (tzv. integritní omezení) a operace s daty. Integritní omezení znamená „starost“ o dodržování pravidel - norem, která se týkají dat v tabulce.

Otcem relačních databází je Dr. Edgar Frank "Ted" Codd (pracovník firmy IBM) který navrhnul RMD a jeho pravidla publikoval v roce 1970 v článku: "The relational model of data for large shared databanks"[9)]. Kromě základních definic byly v článku obsaženy tyto myšlenky:

- Relační model dat odděluje od sebe data, která jsou chápána jako relace, od jejich implementace.
- Uplatnění symetrického přístupu k datům. Při manipulacích s daty nezáleží na přístupových metodách k nim v relacích.
- Pro práci s daty máme k dispozici matematické disciplíny - relační algebru a kalkul, jimiž lze popsat význam slov a konstrukci relačních jazyků.
- Pro omezení nadbytečnosti (redundance) dat máme k dispozici normalizaci relací, což znamená vhodně navrhované databázové struktury.

Základem relačního modelu dat je databázová relace (množina) obsahující data, která se od matematické relace poněkud liší. Model obsahuje tzv. ***schéma relace*** – zde je definováno jméno této relace, jména jednotlivých atributů a popisuje integritní omezení. Každá buňka dat musí odpovídat 1. normální formě.

Relace je tabulka, kde jsou data uspořádaná do sloupců (***atribut*** + ***doména***) a řádků (***n-tic, formát n-rozměrného vektoru***). Databázová relace umožňuje práci s dotazy v tabulkách (není však podporováno ve všech DBMS). Protože relace je množina s omezením ***redundance dat*** (nesmí obsahovat duplicitní prvky) a je uspořádána pouze dvojrozměrně - do sloupců a řádků, neexistuje první, druhý ani n-tý řádek. Řádky nemají specifické pořadí, nejsou proto ani popsatelné číslem řádku, a musí proto existovat nějaká konstrukce, která nám umožní jednotlivé řádky adresovat. Tuto konstrukci nazýváme ***primární klíč***.

Primární klíč je atribut (kód) nebo také soustava atributů, jejichž hodnoty jednoznačně identifikují řádek relace. Databázové servery při definici skupiny atributů jako primárního klíče hlídají jejich jednoznačnost.

Například primárním klíčem v tabulce zaměstnanců bude údaj, který charakterizuje každého člověka - rodné číslo. Sloupce obsahující rodná čísla manželů budou jednoznačně identifikovat manželské páry. Každá relace musí obsahovat primární klíč, v nejhorším (nejsložitějším) případě jím jsou všechny atributy. Jestliže atribut není součástí klíče, nazýváme jej neklíčový.

Základní zásady, které se uplatňují v teorii relačních modelů lze zapsat třemi pravidly:

- Veškerá data se ukládají ve vztazích mezi sloupci a řádky a říkáme jim relace.
- Všechny hodnoty v databázi jsou skalární – každé hodnotě v databázi je přiřazena určitá číselná hodnota (***skalár***).
- Všechny operace se provádějí nad relací a výsledkem jsou jiné relace.

V dnešní době pravděpodobně nenalezneme jakýkoliv systém, který splňuje všechna pravidla absolutně.

Relační databázový systém typu DBMS (Database managment system) je například MySQL[13)]. Jakákoliv manipulace s databázemi, tabulkami či daty provádíme pomocí příkazů, resp. dotazů. Dotazy vycházejí z deklarativního programovacího jazyka SQL (Structured Query Language). Systém MySQL je využitelný v jazyku C, C++, PHP (Hypertext Preprocessor, skriptovací programovací jazyk, určený zejména pro programování dynamických www stránek), Java, Python, Tcl (Tool Command Language), Visual Basic, .NET. MySQL je šířen jako open source, tj. jako software s otevřeným zdrojovým kódem.

1.2.1. Výhody, nevýhody, vývoj a budoucnost

Velkou výhodou relačních databází je přirozená prezentace dat, snadné definování vztahů a zavedení kontroly integrity pomocí omezení. Další velkou výhodou relačních databází jsou např. ***triggery*** či uložené procedury. Trigger (nebo také ***spoušť***) v databázi definuje činnosti, které se mají provést v případě výskytu definované události nad databázovou tabulkou. Například by nám stačilo definovat trigger vázaný na událost update, insert, nebo delete. Tento trigger bude porovnávat záznam jednak před uložením, jednak po uložení a bude také příslušné změny zaznamenávat do tabulky historie.

Uložená (v databázi) procedura je část programu, která je jasně funkčně oddělená od svého okolí, má ***interface*** (seznam parametrů) pro komunikaci s jinými moduly programu. Současně není vyloučeno, aby měla vlastní (lokální) proměnné, které jsou „neviditelné“ pro ostatní části programu. Můžeme se k ní chovat stejně jako k indexu, triggeru či jinému objektu databáze. Můžeme ji založit, upravit i mazat a to pomocí příkazů dotazovacího jazyka - v případě relační databáze za pomoci skupiny příkazů pro definici dat DDL (Data Definition Language) SQL.

Například uložená procedura „Zapiš nového zákazníka“ uloží do tabulky skupiny zákazníků nového zákazníka. Tuto proceduru by mohly volat tři aplikace: internetový obchod, zadání ze zákaznického centra, návštěva kamenné prodejny. Jinak bychom podobnou proceduru museli napsat ve třech verzích - jednou v C++, jednou třeba v Power Builderu a jednou v rámci programu pro internetový obchod (třeba ASP nebo PHP).

Relační databázové systémy jsou velmi dobré pro řízení velkého množství dat, pro vyhledávání dat, ale poskytují poměrně nízkou podporu pro manipulaci s nimi. RDBS jsou založeny na dvourozměrných tabulkách a vztahy mezi daty jsou zde vyjadřovány porovnáváním hodnot v nich uložených. Dotazovací jazyky (včetně SQL) pracují tak, že se tabulky propojí a vyjádří se vztahy mezi daty.

Srovnáme-li relační databáze s objektově orientovaným modelem, který je založen na objektech (struktury, které kombinují daný kód a data), pak objektové databázové systémy umožňují využití hostitelského objektového jazyka jako je třeba C++, Java nebo Smalltalk přímo na objekty "v databázi". To znamená, že nemusíme „přeskakovat“ mezi jazykem aplikace (např. C) a dotazovacím jazykem (např. SQL).

Relační modely dat jsou relativně jednoduché a elegantní a také jsou naprosto odlišné například od objektového modelu.

Relační databáze nejsou databáze, které by byly vhodné pro ukládání objektů. Tady by bylo naprogramování rozhraní pro ukládání objektů v databázi velmi složité.

Relační databáze ukládají data spíše do oddělených tabulek, nikoliv do jednoho velkého úložiště. Zajišťuje to rychlost a flexibilitu.

Relační databázové systémy stále zůstávají nejpoužívanějším typem databázových systémů, i když je pravda, že máme k dispozici v mnoha ohledech pokročilejší objektové databáze. Například aplikační server GemStone/S pro Smalltalk, který poskytuje platformu pro vývoj, provoz a údržbu škálovatelných, vysoce výkonných, vícevrstvých aplikací. Jedna z největších výhod objektových databází je snadná práce s daty ve složitých strukturách, kdežto relační databáze (využívající k ukládání dat plochých relačních tabulek) mají s těmito daty značné potíže.

1.2.2. XML (eXtensible Markup Language, česky rozšiřitelný značkovací jazyk)

XML dokument je vyvinut a standardizován konsorciem W3C a je již sám o sobě někdy považován za databázi, protože poskytuje úložiště dat, dotazovací jazyk (XPath) atd. Proto představa mapování XML dokumentu na tabulku (či tabulky) v relační databázi není příliš obtížná. Výhodou XML je relativně jednoduchá implementace ukládacího mechanismu do relační databáze a také možnost načtení části XML dokumentu. Nevýhodou je, že XML není relační databáze a proto je nutno striktně dodržovat schémata dokumentu či počítat s tím, že není možnost načtení XML dokumentu v rámci jednoho SQL dotazu.

1.2.3. Shrnutí relačních databází

Shrneme-li, u relačních databází je základní výhodou relativně snadná modifikace a propojování tabulek a s nimi spojená možnost dotazů. Naopak mezi její slabiny můžeme zařadit nízkou efektivnost zpracování, které vyžaduje častý přístup na disk, čímž se zpomaluje zpracování.

O relačních databázích se někdy říká, že jsou již za svým zenitem použitelnosti. Já si naopak myslím, že možná teprve teď se začíná využívat naplno jejich potenciál. Myslím si, že jen tak brzy ze světa nezmizí a díky jednoduchosti budou používány ještě hodně dlouho. Jiné technologie budou spíše pro vývojáře, kteří se je budou snažit využívat.

1.2.4. Přehled relačních databází dneška

Nejvýznamnějšími hráči na trhu jsou v současnosti firmy IBM s produktem DB2 a Oracle s Oracle Database, a ty společně ovládají asi 70 % trhu. Dalšími výrobci, i když už s menším podílem na trhu, jsou Sybase a Microsoft s MSSQL. V oblasti webových aplikaci je zřejmě nejvýznamnější opensource databáze MySQL, dalšími OS projekty jsou PostgreSQL nebo Firebird (dříve Interbase) od firmy Borland.

1.2.4.1. Oracle

Firma Oracle byla založena v roce 1997. Její zakladatel Larry Ellison ji pojmenoval Software Development Laboratories. Po dvou letech existence byla společnost přejmenována na Relational Software. Uvedla na trh produkt nazvaný Oracle V2 (i když V1 nikdy na trhu nebyla), která podporuje základní SQL. Jméno Oracle nebylo zvoleno náhodou, ale bylo to údajně krycí jméno projektu financovaného CIA, na kterém Larry Ellison ještě s Bobem Minerem a Edem Oatsem pracovali ve firmě Ampex[7)].

Podle otevřené encyklopedie Wikipedia[8)]: „Oracle je systém řízení báze dat (Oracle database management system – DBMS), moderní multiplatformní databázový systém s velice pokročilými možnostmi zpracování dat, vysokým výkonem a snadnou škálovatelností“.

Velice kvalitním produktem v oblasti databází je platforma Oracle10g, která obsahuje řadu zajímavých technologií. Podporuje databázové zpracování s použitím Gridu, tj. využití výpočetního výkonu libovolného množství geograficky oddělených počítačů. Podstatou datového gridu (mřížky) je sdílení velkého množství dat, poskytování zabezpečeného přístupu k těmto datům a umožnění

jejich následné správy. Tento způsob je založen na vytvoření replikovaných datových katalogů, což vytváří iluzi jednotného, hromadného datového úložiště.

Se zpracováním v gridu souvisí podpora tvorby nepřetržitě dostupných a také vysoce škálovatelných aplikací pomocí technologie Real Application Cluster (RAC), která představuje tzv. shared-disk cluster. Tady je více serverů připojeno ke společnému externímu diskovému prostoru, kde je uložena databáze. Všechny servery mají stejný přístup k datům, přičemž všechny pracují najednou. Jestliže dojde k výpadku některého z nich, ostatní přebírají jeho úlohu.

Podstatná je možnost škálování i na levnějších strojích, které se jen tváří jako clustery a to, že v případě výpadku (plánovaného i náhodného) jednoho či více clusterů nedochází k ovlivnění funkčnosti provozované aplikace.

Mezi další vlastnosti patří například podpora objektových vlastností a XML, bezpečnost zajišťovaná pomocí virtuálních privátních databází, analytické funkce pro datové sklady a dolování dat, či podpora nestrukturovaných dat prostřednictvím Internet File Systém. Platforma Oracle10g je použitelná pro většinu dnešních operačních systémů, Linux a Windows nevyjímaje.

Podporuje dotazovací SQL podle normy SQL92, firemní rozšíření Oracle (například pro hierarchické dotazování), programovací jazyk PL/SQL rozšiřující možnosti SQL (v tomto jazyce můžeme vytvořit například uložené procedury, uživatelské funkce, programové balíky a triggery), mimo to zde můžeme nalézt podporu objektových databází, jež jsou uložené v hierarchickém modelu dat (XML, jazyk XSQL).

Databáze Oracle10g existuje ve čtyřech základních edicích, je zajištěn snadný přechod na vyšší edice v případě zvýšení požadavků zákazníka.

- Oracle Database 10g Express Edition - Bezplatná základní verze, která je určena pro malé firmy, začínající vývojáře a administrátory. I ona však již obsahuje řadu pokročilých funkcí databáze Oracle, jako je například podpora XML, fulltextového vyhledávání nebo práce s prostorovými daty. Verze je k dispozici pro 32bitové operační systémy Windows a Linux. Využije maximálně 1 CPU a 1 GB RAM serveru a umožňuje ukládat až 4 GB dat. Na serveru lze současně provozovat pouze jednu instalaci Oracle Database XE.
- Oracle Database 10g Standard Edition One - Poskytuje všechny služby, které jsou nutné pro efektivní spravování dat a to pro malé podniky nebo jednotlivá

oddělení a pracovní skupiny velkých podniků. Stejně jako všechny ostatní edice může být Standard Edition One použita i pro provoz aplikací pracujících s daty ve formátu dokumentů, XML souborů, s workflow procesy, nebo třeba geografickými či geopolohovými daty.

- Oracle Database 10g Standard Edition - Nabízí funkce obdobné Standard Edition One, s rozšířením o vyšší výkon daný nasazením na servery s kapacitou až čtyř procesorových jader. Nově je doplněna možnost nasazení SE na clustery serverů s celkovou kapacitou do čtyř procesorových jader s využitím technologie Real Application Clusters, která je jinak dostupná pouze jako volitelná a také samostatně licencovaná komponenta k Enterprise Edition.
- Oracle Database 10g Enterprise Edition - Tato edice je špičkou v oboru - umožňuje největším firmám i ve světovém měřítku velmi efektivní provoz podnikových aplikací v konsolidovaném prostředí. Splňuje nejnáročnější požadavky ať již jde o transakční aplikace, datové sklady se složitými dotazy nad velkými objemy dat či internetové aplikace s vysokými požadavky na výkon a spolehlivost. 10g Enterprise Edition není nijak omezena ani z hlediska kapacity ani z hlediska výkonu serverů. Další rozšíření služeb je možné a to pomocí volitelných součástí, které mají samostatné licence a umožňují nasazení databáze v prostředích se specifickými požadavky.

1.2.4.2. IBM DB2

International Business Machines Corporation, se sídlem 1 New Orchard Road, Armonk, New York 10504, USA, je přezdívaná Big Blue neboli Velká modrá, funguje od roku 1888, jako akciová společnost od 15. června 1911.

Na trh byla koncem roku 2006 uvedena devátá verze databáze DB2.

Databázové portfolio produktů DB2 IBM zahrnuje software pro správu dat a informací, který umožňuje ukládání dat, přístup k datům a jejich analýzu v jakémkoliv prostředí.

- DB2 Workgroup Server Edition: Databázový server, který je určený k používání v malých firmách nebo v prostředí jednotlivých oddělení.
- Informix Dynamic Server: Nabízí technologii uložení dat v databázi pro mnohauživatelské prostředí OLTP (online transaction processing) pro výpočetní zpracování v podnicích a pracovních skupinách.

- Informix Dynamic Server Express: Nabízí zabudovatelnou databázi základní úrovně s vysokým výkonem pro malé a středně velké firmy.
- Cloudscape poskytuje lehký, výhradně na Javě založený systém pro správu relačních databází, nebo
- DB2 Express Edition: cenově atraktivní, plně funkční relační databáze, která je dostupná pod operačními systémy Windows a Linux.
- DB2 Enterprise 9: Poskytuje základnu pro datové sklady, zpracování transakcí nebo řešení založená na WWW.
- DB2 Personal Edition: Poskytuje jednouživatelský databázový prostředek ideální pro uživatele PC.

Od začátku roku můžeme 2006 DB2 legálně instalovat zdarma při splnění určitých podmínek. IBM totiž vytvořila balík DB2 Universal Database Express-C (Community Edition), se stejným programovým kódem, jako IBM DB2 Universal Database Express Edition V 8.2, jenomže v menším balíčku. Je určen pro vývojové pracovníky, ale i pro nasazení v menších prostředích. Licence je bez omezení co do počtu uživatelů i co do velikosti databáze. Je ale omezena na max. dva procesory a max. 4 GB RAM[18)].

1.2.4.3. Sybase Adaptive Server Enterprise

Výrobce - společnost **Sybase Software** byla založena roku 1984 Markem Hoffmanem a Bobem Epsteinem v Berkeley v Kalifornii. Sídlí v Dublinu (Kalifornie) a je dnes největší softwarovou společností na světě, výhradně zaměřenou na správu a přenos informací z datových center až do míst jejich využití.

Databázový systém Sybase je založen na původním systému INGRES s rozšířením o SQL rozhraní. Sybase byla do druhé poloviny osmdesátých let 20. století druhou nejpoužívanější databází po Oraclu. V tomto období se firmy Microsoft a Sybase dohodly na sdílení zdrojového kódu a obě tak vyvíjely téměř shodný systém.

SQL Server byl uveden pod OS/2. V této době, tj. kolem vzniku verze 4.9, se Sybase a Microsoft neshodly na další spolupráci a každá firma se vydala jinou cestou. Firma Microsoft se soustředila na svoje produkty, Sybase se soustředila na svoje původní platformy. Sybase postupně ztrácela své pozice na trhu a v současnosti je její produkt až za třemi lídry trhu – za Oraclem, IBM DB/2 a MS

SQL Serverem, který paradoxně vychází ze stejných základů. Až s novými produkty (například SQL Anywhere Studio) se podařilo společnost zachránit.

Nosným produktem v oblasti databází je v současnosti pro Sybase robustní platforma Sybase Adaptive Server Enteprise (Sybase ASE), která podporuje vedle běžně dostupných prvků (jazyk SQL, transakční zpracování, snadná administrace) ne příliš časté vlastnosti - například podporu zpracování transakcí v heterogenním databázovém prostředí či ladění příkazů v jazyce SQL, o které se stará SQL Debugger. Sybase ASE také může nabídnout podporu pro automatické průběžné přizpůsobování data serveru aktuálním požadavkům. Využívá například dynamicky optimalizovaný výkon, je možno měnit výkon serveru bez nutnosti jeho zastavení a tedy za plného provozu.

Podporovanými operačními systémy jsou zejména kvalitní unixy (HP-UX, IBM AIX apod.), i operační systémy společnosti Microsoft i když serverová část vyžaduje řadu NT.

Nejnovější verze jsou založeny na tomto základu a přinášejí inovace zaměřené na e-Business s efektivní podporou datových typů charakteristických právě pro tuto oblast. Jde o někdy velmi dynamické změny výkonnostních parametrů internetových aplikací, požadavek na zvýšenou bezpečnost pro maximální ochranu dat.

Uvedená vylepšení poskytují zákazníkům velmi robustní a výkonnou platformu pro správu dat, která umožňuje snadný přechod do oblasti e-Businessu, zrychlení vývoje těchto aplikací, ochranu kritických dat a usnadnění administračních procesů.

Podle materiálů Sybase cituji: „Inovace ASE jsou cíleny zejména na následující klíčové oblasti: pokročilá správa dat pro e-Business, dynamická optimalizace výkonu a bezpečnost pro e-Business“[6].

Je to systém, který může být nasazen v prostředí, které má vysoké nároky zejména na bezpečnost, stabilitu, spolehlivost a bezpečnost dat a vysoký výkon. Nemá vysoké systémové nároky, je otevřený, podporuje nepřetržitou dostupnost a škálovatelnost a tím je předurčen i pro podniková portálová řešení.

Například podnikový databázový systém Sybase ASE (Adaptive Server Enterprise) splňuje bezpečnostní certifikát podle kritérií EAL4 (Common Criteria - ISO 15 408) tj. současný nejvyšší dosažitelný stupeň certifikace.

2. Přehled jednotlivých verzí MS SQL 2000 a 2005

Firma Microsoft již nedoporučuje použití MS SQL 2000 nad operačním systémem Windows Vista, nebo Windows Server „Longhorn“ a to hlavně z důvodů bezpečnosti. Všeobecně se také doporučuje přechod z verze 2000 na příslušné verze MS SQL 2005. Nebudeme se zde zmiňovat o hardwarové náročnosti, o kterých si každý zájemce může dočíst na webových stránkách firmy Microsoft.

2.1. Jednotlivé verze MS SQL 2000

MS SQL 2000 je stále hojně používaný databázový stroj a my se seznámíme s jeho jednotlivými edicemi. Probereme si jejich dostupnost a jednotlivá omezení[2)].

2.1.1. Verze Desktop Engine Edition

Tato edice je a byla vždy dostupná zdarma, dnes už raději doporučujeme použít verzi MS SQL Server 2005 Express edition, která je také zdarma. Desktop Engine Edition obsahuje pouze hlavní relační databázový systém a neobsahuje administrativní nástroje, jako jsou Enterprise Manager a Query Analyzer.

2.1.2. Verze Personal Edition

Jak už z názvu vyplývá, jedná se o verzi Personal nebo-li osobní. Je velmi vhodná pro samostatné pracovníky, kteří jsou „odříznuti“ od centrální databáze. Po připojení k síti můžeme data synchronizovat pomocí mechanizmů replikace s hlavní databází.

Tato verze může být dále využita také jako samostatný malý server. Podporuje činnost více procesorů (multiprocessing), ovšem na druhou stranu nepodporuje paralelní dotazy (během kterých se zpracovávají různé části stejného dotazu). Samostatně se zakoupit nedá, dodávala se jako součást edice Standard nebo Enterprise.

2.1.3. Verze Standard Edition

Tato edice je nejdůležitější variantou SQL Server 2000. Podporuje práci více procesorů (multiprocessing), u této verze to znamená práci až čtyř procesorů a až 2 GB paměti RAM. Využívá pouze omezenou část funkcí služeb Analyzer Services. Pro každý počítač s instalovanou verzí Standard Edition je nutné koupit licenci.

2.1.4. Verze Developer Edition

Tato verze obsahuje prakticky všechny funkce verze Enterprise Edition, jediný rozdíl je v licenci. Edice Developer Edition je určena pouze pro vývoj a testování, po dokončení této fáze by měl nastat přechod na práci v Enterprise Edition.

2.1.5. Verze Enterprise Edition

U této verze můžeme využít práci až 32 procesorů, ale také podporu clusteru (umožňuje nám vzájemně propojit dva servery, které se navzájem jistí proti případné havárii a v průběhu běžného provozu si zatížení rozdělují). Možnost clusteru je dostupná pouze u této verze a nejen to se zohlednilo i v ceně licence, která byla několikrát větší než u verze Standard. Jako další výhodu této vrze můžeme považovat podporu více jak 4 GB RAM, indexované pohledy, nebo např. záznam protokolu nebo-li možnost zotavení po havárii.

2.2. Jednotlivé verze MS SQL 2005

Podobně jako u verze 2000, se i zde seznámíme s jednotlivými verzemi MS SQL 2005. Probereme si jejich použití, aplikaci, popřípadě novinky jednotlivých edicí[17)].

2.2.1. SQL Server 2005 Express Edition

Velmi oblíbený produkt firmy Microsoft, na kterém si zde v další části ukážeme některé příklady. Je určena hlavně pro začátečníky a zájemce, kteří se chtějí seznámit s MS SQL 2005. Její obliba tkví hlavně v její dostupnosti, neboť je zcela zdarma a na rozdíl od verze MSDE (SQL 2000) nemá omezen počet uživatelů.

Mezi její omezení ovšem patří:

- velikost databáze (maximálně 4 GB prostoru na disku),
- využití maximálně 1 procesoru a 1 GB operační paměti serveru.

Má-li server větší operační paměť nebo více procesorů, Express Edition využije server pouze do výše svého omezení.

2.2.2. SQL Server 2005 Workgroup Edition

Dostupná edice hlavně pro malé a střední podniky. Podporuje maximálně dva procesory, velikost databáze je neomezená a operační paměť může mít až 2GB. Tuto edici můžeme považovat za relativně levné řešení pro menší společnosti, kterým již nevyhovuje předchozí verze Express Edition, ale současně nechtějí investovat nemalé peníze do následujících verzí Standard a Enterprise Edition.

2.2.3. SQL Server 2005 Standard Edition

Edice určená hlavně pro střední podniky. Pracuje s maximálně čtyřmi procesory, neomezenou velikostí jak databáze, tak velikostí operační paměti. Obsahuje některé funkce, které jsme ve verzi 2000 nalezli pouze v edici Enterprise.Edition.

Standard Edition obsahuje mimo jiné:

- ***SQL Server Integration Services*** - nástroj, který nám pomůže získat různá data z různých aplikací, jako například Excel, textový soubor apod., transformovat je, zkontrolovat a následně uložit do datového skladu.
- ***SQL Server Analysis Services*** - komplexní a integrovaná služba pro data mining, analyzování a reporting.
- ***SQL Server Reporting Services*** - služba pro vytváření sestav, která nám umožní tiskovou, ale i interaktivní webovou prezentaci dat.

Tyto nástroje se dohromady nazývají nástroje „***Business Inteligence***".

2.2.4. SQL Server 2005 Enterprise Edition

Komplexní řešení pro velké organizace a společnosti. Nabízí všechny vymoženosti předchozí verze (Standard edition), ale také mnohé funkce navíc:

- rozšířené možnosti ***Business Inteligence***,
- ***zrcadlení databáze*** - pokročilá funkce pro zotavení po havárii, která umožňuje kopírování transakčního protokolu na druhý server. Umožňuje tak rychlé přesunutí provozu na druhý server,
- ***partitioning*** - tato funkce umožňuje fyzické rozdělení tabulek, jejichž velikost přesahuje možnosti použití systémových prostředků. Například tabulku zákazníků na tabulky rozdělené dle začátečního písmene jména zákazníka.
- ***databáze Snapshot*** - pouze Enterprise verze nám poskytuje jinou možnost obnovy databáze, než jakým je obnova dat ze zálohy. Používá se v případě potřeby obnovy menších částí, které byly smazány nebo ztraceny, popřípadě špatně aktualizovány. Během tohoto procesu může databáze zůstat on-line, bez nutnosti odpojení od sítě. Tato možnost je velmi užitečná zejména pro velké databáze, jejichž obnova ze zálohy by trvala nepřiměřeně dlouho a odpojení od sítě by bylo nežádoucí. Snapshot je pouze Read-only (můžeme z něho pouze číst) snímek databáze v daném čase. Při vytvoření je Snapshot prázdný a plní se od okamžiku jakékoliv změny ve zdrojové databázi. Tato technologie nejprve uloží původní data z databáze do „Snapshotu" a následně upraví zdrojovou databázi. Do Snapshot databáze se tudíž dostanou pouze data, která byla změněna a v případě, že nedochází k časté aktualizaci databáze, je Snapshot mnohem menší než základní databáze. Snapshot se také často programuje tak, aby se spouštěla jeho nová (prázdná) verze vždy v určitý čas (například vždy po 3 hodinách).

2.2.5. SQL Server 2005 Developer Edition

Tato verze je určena zejména pro vývojáře k tvorbě a testování aplikací na platformách 32-bit, ia62 a x64. Obsahuje všechny prostředky a komponenty, které obsahuje verze Enterprise Edition, je ovšem licencována pouze pro vývoj, testování a zkoušení demoverzí. Následně je možné tuto verzi, po testování a zkoušení, velmi jednoduše rozšířit na SQL Server 2005 Enterprise Edition.

2.2.6. SQL Server 2005 Compact Edition

Nová generace mobilních zařízeních a příslušné aplikace měly za následek vývoj další verze SQL Serveru. Vývoj nejen PDA (Personal Digital Assistant – Osobní digitální pomocník), přenosných přehrávačů, ale i herních konzol zapříčinil zájem Microsoft o vývoj této verze MS SQL 2005. Software i podpora je zdarma pro jednouživatelské klientské aplikace pro všechny platformy Windows, včetně Tablet PC, Pocket PC, Smart Phone ale i pro klasické desktopy.

2.2.7. SQL Server na platformách 64-bit

Mezi další vlastnost u některých verzí SQL Serverů verze 2005 můžeme považovat rozšíření a optimalizaci práce na serverech typu x64 nebo Itanium a na Microsoft Windows Server 2003. Tuto vlastnost můžeme nalézt ve verzích Standard, Enterprise a Developer.

3. Přístup k datům ve verzi MS SQL 2005

Na začátku práce s daty byla jedna aplikace, která obsahovala jak programový kód tak samotná data. To bylo velmi nepraktické hlavně z důvodu nutnosti kompilace celého programu v případě jakékoliv změny dat.

Následovala monolitická aplikace s daty v externím souboru. Tato varianta již měla oddělená data od samotného programového kódu a byla určena zejména pro práci jednoho uživatele. Díky tomu mohl uživatel pracovat s daty bez nutnosti kompilovat celý program.

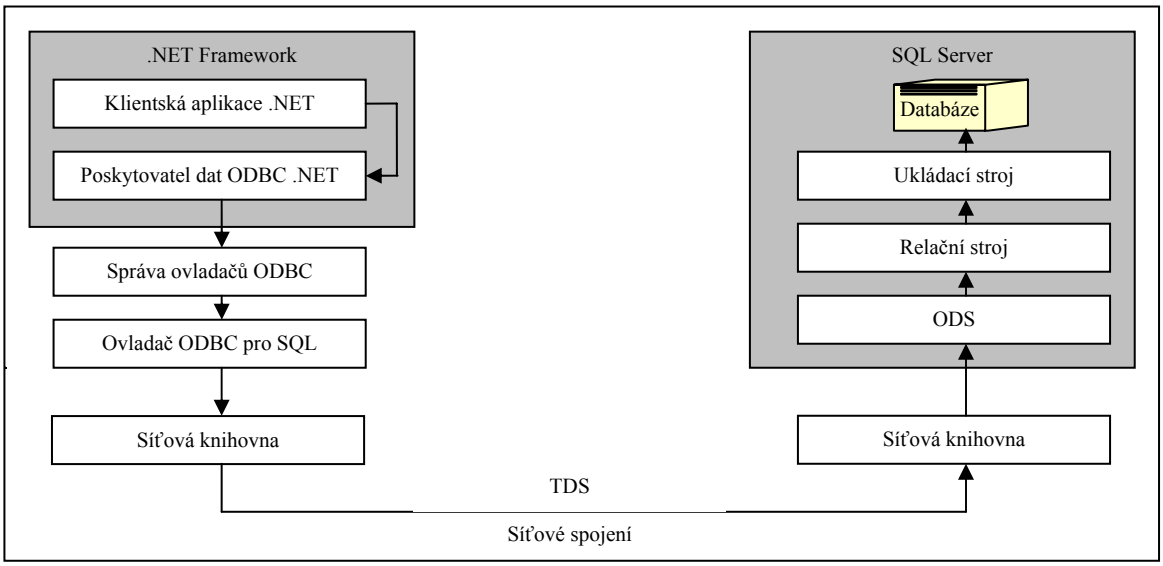
Dalším krokem bylo zpřístupnění souboru s daty více uživatelům. Tito uživatelé mohou data využívat pomocí různých aplikací a problém nastává v případě, že tito uživatelé budou požadovat přístup s právem změny v databázi (ať už mazání, zapisování nebo úpravy dat). Zde musí tvůrce databáze nastolit jasná transakční pravidla pro práci s daty. Tyto pravidla mohou být velmi složitá a bývá velmi obtížné je dodržet.

Následovala myšlenka tvorby samostatné služby, která bude samostatně rozhodovat o právu přístupu k datům, čímž předejde konfliktům mezi požadavky jednotlivých aplikací. Všechny aplikace potom musí bez výhrady přistupovat k databázi pomocí této služby.

V současnosti existuje několik rozhraní, které poskytují jednotný přístup do databázových systémů od různých výrobců. Tato rozhraní napomáhají programátorům k vytváření aplikací, které komunikují se serverem. Probereme si nejznámější z nich, které využívá MS SQL 2005.

3.1. ODBC (Open Database Connectivity)

ODBC můžeme obecně považovat za prostředek, který náš dotaz SQL přenese až přímo k databázi. Toto rozhraní je vytvořeno pro použití nezávisle na programovacím, operačním a databázovém systému. V praxi se velmi osvědčilo použití zejména pro programovací jazyk C++, ale také Java, Virtual Basic, nebo například ve skriptovacích jazycích PHP, ASP atd.



Obrázek 1 - Připojení k serveru MS SQL 2005 pomocí ODBC[1)]

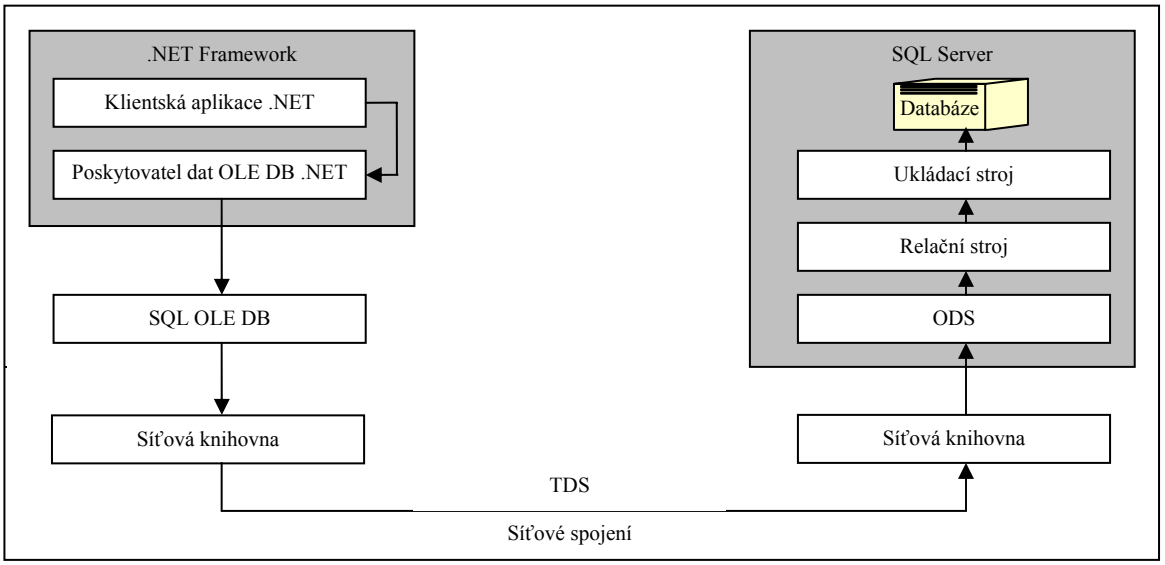
Nejprve vysvětlíme některé pojmy z obrázku:

- **.NET Framework:** Aplikace vytvořené v platformě .NET mají stejné hlavní rysy a model návrhu je stejný jak pro přenosná zařízení, desktopy, servery ale také pro vytváření www. Jádrem platformy .NET je .NET framework. Aplikace postavené na této platformě můžeme považovat za RUN-TIME, nebo-li aplikace vytvořené pro toto prostředí, jsou s ním pevně svázány a nelze je bez tohoto rozhraní spustit. Toto jádro nám poskytuje usnadnění například při práci s databázemi a datovými zdroji.

- **ODS (Operation Data Store):** Nebo-li sklad provozních dat. Databáze, která podporuje sledování provozu a obsahuje komplexnější pohled na data v provozních systémech. Tuto databázi můžeme také považovat za mezikrok pro aplikace typu **OLTP** (On-Line Transaction Processing – technologie pro uložení dat v databázi způsobem, který klade důraz na snadné a bezpečné ukládání jakýchkoliv změn v datech) a **OLAP** (On-Line Analytical Processing – klade důraz na ukládání velkých objemů dat tak, aby bylo možné jednoduše získat data potřebná pro analýzy dat, trendů a výsledků).
- **TDS (Tabular data stream):** Protokol, nebo také pravidla pro přenos dat mezi dvěma počítači. Vytvořeno firmou Sybase Inc. a následně implementováno firmou Microsoft pro Microsoft SQL Server založeném na kódu Sybase.

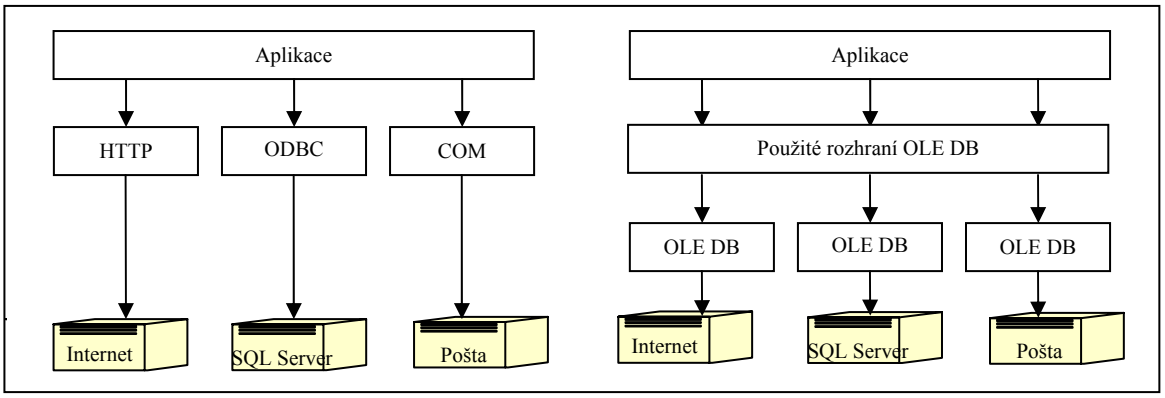
3.2. OLE DB

OLE DB můžeme nazývat množinu rozhraní, které přistupují nejen k databázím, ale i k webovému nebo poštovnímu rozhraní. Jedná se o jakési UDA (Universal data access) rozhraní. Strukturu a ukázku použití OLE DB a ODBC rozhraní můžeme vidět níže.



Obrázek 2 - Připojení k serveru MS SQL 2005 pomocí OLE DB[1)]

- **OLE DB poskytovatel:** rozhraní, které přistupuje přímo do databáze a jejím datům, popřípadě přistupuje k dalšímu rozhraní, které má již přímý přístup do databáze. Ve většině případů je součástí databázového jádra. Například C++ obsahuje poskytovatele pro MS Access, SQL Server, Oracle nebo ODBC.

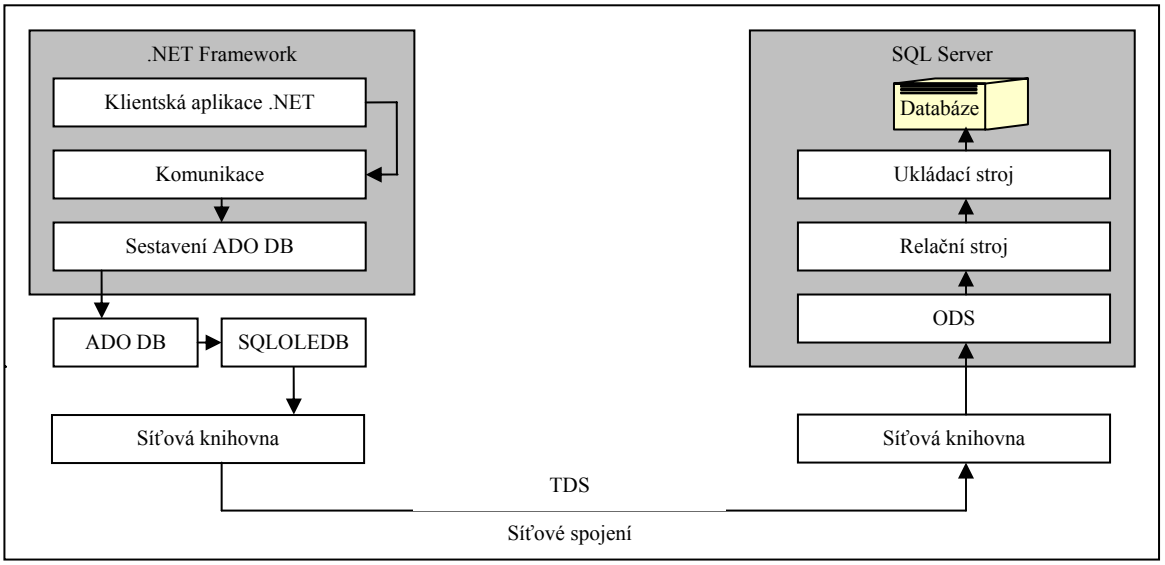


Obrázek 3 - OLE DB a ODBC - Přístup aplikace pomocí různých rozhraní

Jak vidíme na levém příkladu u předešlého obrázku, aplikace musí použít pro přístup k databázi, poště nebo podobným zdrojům dat různá rozhraní. Ve druhém případě přistupuje aplikace ke všem zdrojů pomocí jednoho rozhraní.

3.3. ADO.NET

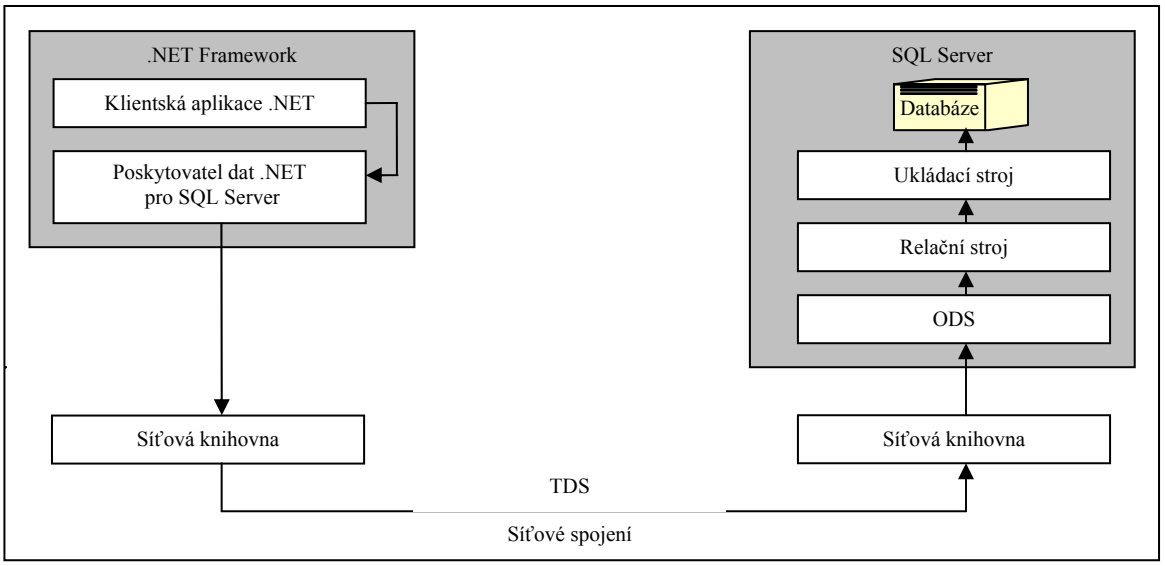
ADO (Active Data Objects) pro technologii .NET. Tato technologie nabízí SQL Serveru 2005 poskytovatele řízených dat, tím odpadá nutnost potřeby externího datového ovladače či poskytovatele dat. Toto rozhraní komunikuje přímo s databázovým serverem bez jakékoliv programovatelné vrstvy. Můžeme také říci, že ADO.NET je rozhraní (třídy), které používáme pro přístup k datovým zdrojům v různých aplikacích .NET framework. Na rozdíl od ADO může být také použita pro odpojené aplikace. ADO.NET také podporuje formát XML.



Obrázek 4 - Připojení k serveru MS SQL 2005 pomocí ADO.NET[1)]

3.4. SQL Server Native Klient

Pro rychlé zpracování a snížení režie, jež může přinášet technologie .NET, vyvinul Microsoft pro SQL Server 2005 nového klienta SQL Server Native Client. Jak můžeme vidět na obrázku níže, tento klient se připojuje k SQL Serveru přímo, bez použití dalšího rozhraní typu ODBC, nebo OLE DB.



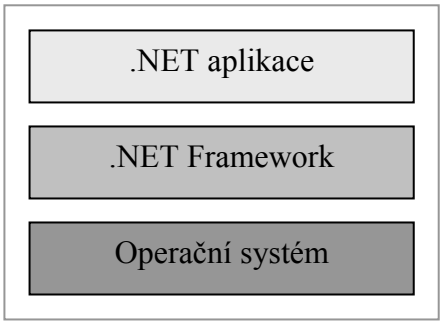
Obrázek 5 - Připojení k serveru MS SQL 2005 pomocí SQL Server Native Client[1)]

4. Struktura SQL 2005 a .NET Framework

V předchozí kapitole jsme se dotkli názvu .NET Framework. Tato nadstavba na operační systém byla implementována přímo do samostatného jádra SQL Serveru 2005 a proto si rozebereme trošku podrobněji jak .NET Framework, tak i následně strukturu MS SQL 2005.

4.1. .NET Framework

Hlavně podpora internetových a síťových aplikací dala pravděpodobně jméno .NET této nadstavbě nad operační systém. Níže můžeme vidět hierarchický vztah .NET Framework k operačnímu systému.

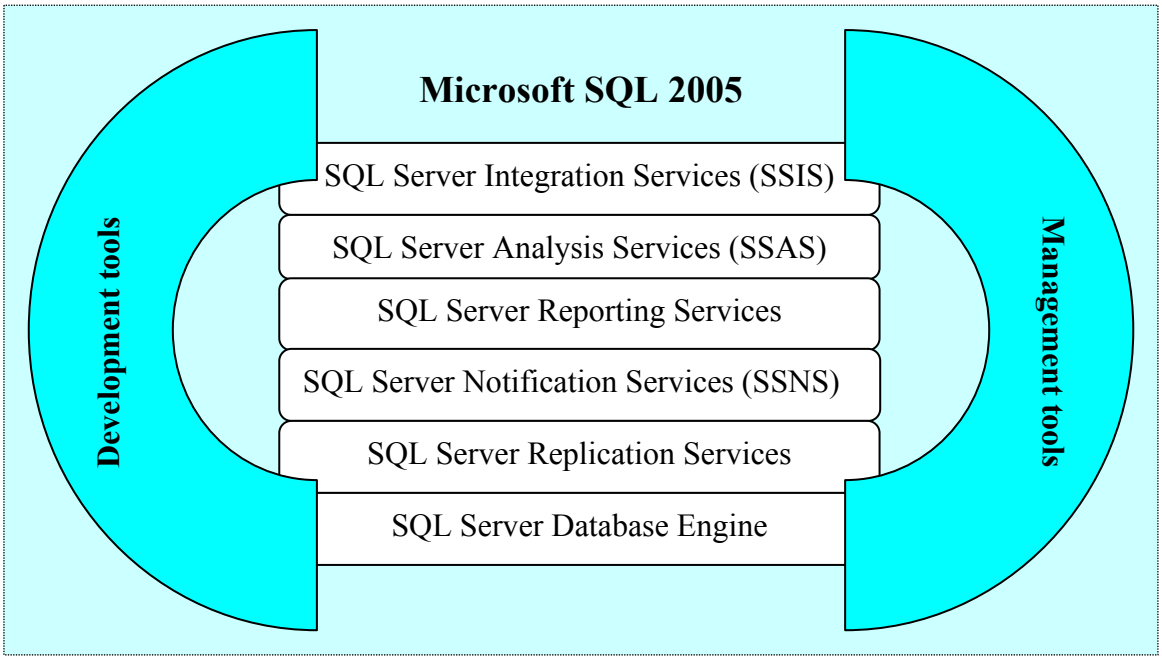


Obrázek 6 - .NET Framework

Bez ohledu na programovací jazyk typu .NET (například Visual C#, Visual Basic, Visual J#), můžeme používat například stejné datové typy. Tento Framework nám nabízí stejné rozhraní pro všechny programovací jazyky typu .NET. Zvládneme-li jeden z těchto programovacích jazyků, není problém přechodu na jiný, pouze syntaxe bude jiná. Tento rámec si také můžeme představit jako systém knihoven, který nabízí jednotné prostředí pro vyšší programovací jazyky.

4.2. Struktura SQL 2005

Jako každá databáze tak i SQL 2005 musí řešit spoustu různorodých úkolů. Zabezpečení podnikových dat, poskytovat managementu podporu v rozhodování (BI – Business Inteligence), poskytovat data pomocí různých rozhraní, umožnit snadnou manipulaci s daty včetně jejich analýzy, ale také nabízet přívětivé prostředí pro vývojáře. Nejen pro tyto úkoly má SQL 2005 následující prostředky (viz obrázek), které si postupně rozebereme.



Obrázek 7 - Struktura SQL 2005[16)]

4.2.1. SQL Server Integration Services (SSIS)

Tato služba je pro SQL 2005 poměrně nová, přebírá funkci transformace dat z verze 2000 (DTS – Data Transformation Services) a samozřejmě jej podstatně rozšiřuje. Tuto službu můžeme také popsat jako „Vytváření aplikací pro integraci dat na úrovni podniku“.

Umožňuje nám získat, transformovat a ukládat data z různých zdrojů (textový soubor, Excel, XML, Oracle atd.) do námi požadované podoby. Můžeme je také zkontrolovat, očistit a uložit do databáze nebo datového skladu. S takto získanými zdroji můžeme nakládat, bez ohledu na to, kde jsme je získali.

SSIS se snaží provádět různé transformace a čištění dat v paměti a tím minimalizovat mezi-ukládání. Oproti předchozí verzi 2000 zde můžeme nalézt následující vylepšení:

- **Podmíněné rozdělení dat:** určíme si podmínku, na základě které se rozdělují přijímaná data. Například zákazníky do jednotlivých tabulek dle abecedy, pomocí začátečního písmene jejich příjmení.
- **Data conversion:** konverze dat mezi různými typy (číselnými, řetězci atd.).
- **Aggregate**: vykoná vícenásobnou agregaci do jednoho průchodu.
- **Sort**: třídí data do jednolitého toku.
- **Merge, Merge Join a UnionAll**: může vykonat operace slučování.
- **Derived Column**: odvozené sloupce.
- **Audit**: přidá sloupce s počtem řádků a jiná metadata.

4.2.2. SQL Server Analysis Services (SSAS)

Dnes nejrozšířenější OLAP (On Line Analytical Processing) produkt. Využívá se pro rychlou a hloubkovou analýzu rozměrných databází a datových skladů, k vytváření sestav, zobrazení různých přehledů, výkonů atd. Navazuje na verzi 2000 a samozřejmě i zde nalezneme podstatná vylepšení:

- **XML for analysis:** pomocí této služby může Analysis Services fungovat jako webová služba. XMLA můžeme nazvat také protokolem pro komunikaci se severem.
- **Translations:** kostky OLAP můžeme vytvářet ve vícejazyčné verzi a jejich uživatelé si mohou zvolit jazykovou mutaci, ve které chtějí vytvářet dotazy na samotnou kostku.
- **Data Mining:** nástroj pro dolování dat. Pomocí tohoto prostředku můžeme předvídat události, které nastanou, a proč se dané události dějí.

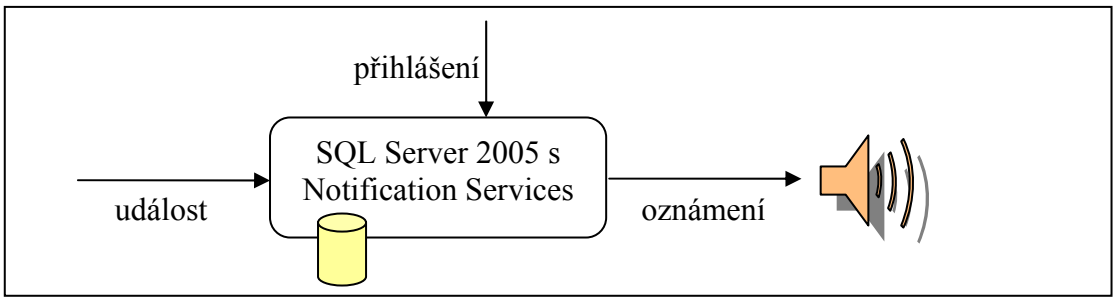
4.2.3. SQL Server Reporting Services

Velmi užitečný nástroj pro tvorbu, správu, a posílání sestav ve formě papírové, interaktivní nebo pomocí webových sestav. Samozřejmostí je provázání s Microsoft Windows Server a Microsoft Office.

Mezi její výstupy můžeme zahrnout zejména aplikace MS Office, PDF apod. Pomocí rozhraní .NET Framework se můžeme připojit k různým zdrojům dat a následně vytvářet další výstupní formáty.

4.2.4. SQL Server Notification Services (SSNS)

SSNS je struktura pro vytvoření aplikace, která vytvoří a následně zašle upozornění. Schéma událostí můžeme vidět níže:



Obrázek 8 - SQL Server Notification Services (SSNS)

Po spuštění aplikace a přihlášení se SSNS průběžně sleduje zprávy a dané události. Poté co nastane specifikovaná událost (např. teplota v reaktoru dosáhne kritické hodnoty), SSNS nalezne shodu mezi kritickou hranicí a hodnotou pro zaslání upozornění a patřičně zareaguje (například zašle příkaz pro spuštění hlasového souboru s vyhlášením krizového stavu).

Aby bylo upozornění zasláno pouze jednou, existuje mechanizmus, který v případě potřeby zabrání opakovanému upozornění.

SSNS umí zasílat zprávy v různých formátech, například uložit do souboru nebo zaslat email (pomocí SMTP serveru). Ve verzi MS SQL 2005 Enterprise můžeme také nastavit zasílání upozornění pro více příjemců.

4.2.5. SQL Server Replication Services

Služba pro kopírování a distribuci dat a databázových objektů z jedné databáze do druhé. Pomocí replikace můžeme šířit data do různých odpojených nebo mobilních zařízení. Můžeme také integrovat různé nestejnorodé systémy včetně integrace dat z databází Oracle.

4.2.6. SQL Server Database Engine

Jádro celého SQL Server 2005 pro ukládání, manipulaci a bezpečnost dat.

4.2.7. Development tools

Nástroje pro vývojáře, které jsou spjaty s Microsoft Visual Studio a dovolují nám manipulaci s daty, data mining, ale také práci s funkcemi OLAP a následný vývoj aplikací.

4.2.8. Management tools

Nástroj pro správu a ladění databází, který umožňuje úzkou spolupráci s **MOM** (Microsoft Operations Manager – sleduje a řídí stav, výkon a úroveň zabezpečení serveru a jednotlivé aktivní prvky. Monitoruje stav jednotlivých prvků a vytvoří upozornění v případě potřeby, identifikuje a lokalizuje problémy. Dokáže generovat reakci, která se automaticky provede, ale také spolupracovat s **SMS** (Microsoft Systems Management Server – nástroj pro správu a konfiguraci aplikací, softwaru a operačních systémů ve vnitropodnikových sítích). Obsahuje podporu webových služeb a jiných heterogenních aplikací, ale také snadnou integraci stávajících dat se SQL Serverem.

5. T-SQL a nové možnosti ve verzi MS SQL 2005

V této kapitole se budeme věnovat novým možnostem dotazovacího jazyku MS SQL 2005, neboli Transact SQL (T-SQL).

5.1. Co je T-SQL

Také jazyk SQL (Structure query language – strukturovaný dotazovací jazyk) prošel svým postupným vývojem a byl standardizovaný jak dle ANSI tak dle ISO (první standardizaci proběhla v roce 1987 – SQL 86 a poslední v roce 2006 – SQL 2006). Tato standardizace vedla jak ke kompatibilitě tak obecně k přehlednosti SQL. Každá vývojářská firma si ovšem nad tento standard přidává svá vylepšení, kterými se snaží zdokonalit svůj produkt a předběhnout konkurenci. U Microsoft a jeho MS SQL 2005 (a příslušné starší verze), je to implementace jazyku Transact-SQL.

Samotné slovo „Transact“ (Transakce) označuje způsob práce databáze s jednotlivými dotazy. Transakce je dále nedělitelná a uzavřená operace, která musí proběhnout buď celá, nebo vůbec. Tato vlastnost má velký vliv na konzistenci databáze. Pokud dojde během průběhu transakce k chybě (jako například k výpadku systému) a tato operace nebyla řádně dokončena, systém má možnost vrátit se zpět do konzistentního stavu, který byl před započetím transakce. Myšlenkou a způsob jak T-SQL pracuje je, že před samotným

vykonáním operace (transakce), jsou všechny příkazy vykonány databázovým strojem, mohou být načteny do vyrovnávací paměti a zvolen optimalizovaný postup. Databázový stroj je schopen zhodnotit jednotlivé příkazy ještě před jejich samostatnou fyzickou implementací a v případě, že nalezne chybu, může započatou transakci vrátit zpět do původního konzistentního stavu. Samozřejmě tímto způsobem práce s databází dochází ke snížení výkonu ve srovnání s přímým fyzickým zápisem do databáze. Je to ovšem bohatě vyváženo stabilitou systému.

5.2. Nejvýznamnější novinky v T-SQL 2005

Zde si probereme nejvýznamnější novinky v příkazech T-SQL 2005 a jednotlivé funkce si ukážeme na příkladech.

5.2.1. Příkaz „TOP n“

Z názvu jednoznačně vyplývá, že se jedná o příkaz, který nám z vybrané množiny vybere pouze „n“ záznamů. Tento příkaz existoval již ve starších verzích, nyní ve verzi 2005 ovšem můžeme zadat hodnotu parametru n pomocí proměnné, čímž lze tuto hodnotu libovolně měnit. O tom, jestli budou tyto hodnoty nejvyšší nebo nejnižší, rozhodneme pomocí příkazu ***ORDER BY*** („starý známý“ příkaz pro třídění).

Následuje jednoduchá ukázka pro vysvětlení příkazu TOP n:

Zkušební tabulka ***Zamestnanci***:

ID	Prijmeni	Jmeno	Prescasy
25	Tomas	Vavrina	22
41	Ondra	Soukup	16
42	Radek	Morkes	19
52	Bara	Kuklova	15

tabulka 12 - TOP n – zadání

Následuje samotný kód:

```
DECLARE @n int
SET @n=2
SELECT TOP(@n)  ID,
                Prijmeni,
                Jmeno,
                Prescasy
FROM Zamestnanci
ORDER BY Prescasy DESC
```

Vysvětlíme si celý tento kratičký příklad (samozřejmě již musí existovat výše uvedená tabulka Zamestanci). Nejprve si deklarujeme proměnnou ***n*** jako celé číslo, následně nastavíme tuto proměnou na hodnotu 2. Dále příkazem ***SELECT*** a ***TOP (@n)***, vybereme 2 zaměstnance, kteří mají největší počet přesčasů. Že se jedná o zaměstnance, kteří mají nejvíce přesčasových hodnot, rozhodujeme příkazem ***ORDER BY*** nastavenou na hodnotu ***DESC*** (řazení sestupné).

Zde naleznete výstup výše uvedeného dotazu:

ID	Prijmeni	Jmeno	Prescasy
25	Tomas	Vavrina	22
42	Radek	Morkes	19

tabulka 13 - TOP n – výsledek

5.2.2. Příkazy RANK, DENSE RANK, ROW_NUMBER a NTILE

Na jednom příkladě si ukážeme funkci hned čtyř funkcí, které jsou si svým základem velmi podobné, každá samozřejmě má jiné výstupy. Po ukázce příkladu si jednotlivé funkce rozebereme.

Příklad nových funkcí RANK, DENSE RANK, ROW_NUMBER a NTILE:

Nejprve si vytvoříme tabulku ***Hraci***, dle níže uvedeného příkladu, se kterou budeme pracovat:

Hrac_ID	Klub_ID
235	2
325	2
481	1
569	2
578	2
612	2
719	3
825	3

Hrac_ID	Klub_ID
952	2
1025	1
1139	2
333	3
222	1
123	2
321	3
145	2

Hrac_ID	Klub_ID
258	1
852	2
654	3
456	2
987	1
789	2
159	3
951	2

tabulka 14 - RANK, DENSE RANK, ROW_NUMBER, NTILE – zadání

Následuje jednoduchá ukázka pro vysvětlení jednotlivých příkazů:

```
DECLARE @n int
SET @n=10
SELECT TOP (@n) Hrac_ID, Klub_ID,
ROW_NUMBER() OVER (ORDER BY Klub_ID) AS ROW_NUM,
RANK() OVER (ORDER BY Klub_ID) AS RANK,
```

DENSE_RANK() OVER (ORDER BY Klub_ID) AS DEN_RANK,
NTILE(10) OVER (ORDER BY Klub_ID) AS NTILE
FROM Hraci
ORDER BY Klub_ID

Nejprve pomocí již dříve uvedené funkce ***TOP n*** vybereme pouze „top“ 10 hráčů seřazených dle ***Klub_ID*** (ASC implicitně zadáno u příkazu ***ORDER BY***). Následuje samotné jednoduché použití jednotlivých příkazů, jejich výsledky si vysvětlíme na následující tabulce.

Zde naleznete výstup výše uvedeného dotazu:

Hrac_ID	Klub_ID	ROW_NUM	RANK	DEN_RANK	NTILE
235	1	1	1	1	1
325	1	2	1	1	1
481	1	3	1	1	2
569	2	4	4	2	2
578	2	5	4	2	3
612	2	6	4	2	3
719	3	7	7	3	4
825	3	8	7	3	4

tabulka 15 - RANK, DENSE RANK, ROW_NUMBER, NTILE – výsledek

Nyní si probereme jednotlivé výsledky:

„ROW_NUMBER ( )“: jak již jeho překlad do českého jazyka naznačuje, udává nám pořadí řádku dle zadaného kritéria.

„RANK ( )“: pomocí tohoto příkazu můžeme ohodnotit jednotlivé záznamy podle různých kritérií a vyjádřit jejich pořadí. Na rozdíl od následujícího příkazu přeskakuje hodnoty.

„DENSE_RANK ( )“: chová se stejně jako příkaz ***RANK ( )***, pouze nepřeskakuje hodnoty.

„NTILE ( )“: rozdělí celkový počet řádek na přesně specifikovaný počet tříd. Tento počet zadáme do závorek příkazu.

5.2.3. Příkaz pro ošetření chyb TRY CATCH

Velmi významná funkce pro ošetření chyb, která umožní pomocí bloků ***BEGIN TRY – END TRY*** a ***BEGIN CATCH – END CATCH*** ošetřit celé bloky. V dřívějších verzích MS SQL bylo nutné ošetřit možnou chybu každého příkazu.

Následující příklad ukáže ošetření chyby a následné uvedení informace o stavu transakce:

Vytvoříme si dvě tabulky:

```
CREATE TABLE centralni_sklad
( hodnota int NOT NULL PRIMARY KEY )
CREATE TABLE nizsi_uroven_skladu
(hodnota int NOT NULL REFERENCES centralni_sklad (hodnota))
```

Do centrálního skladu vložíme hodnoty:

```
INSERT INTO centralni_sklad VALUES (101)
INSERT INTO centralni_sklad VALUES (202)
INSERT INTO centralni_sklad VALUES (303)
```

Vložíme do druhé tabulky hodnoty a ošetříme případ, že hodnota není uvedena v první tabulce:

```
BEGIN TRY
BEGIN TRAN
INSERT INTO nizsi_uroven_skladu VALUES (101)
INSERT INTO nizsi_uroven_skladu VALUES (202)
INSERT INTO nizsi_uroven_skladu VALUES (404)
COMMIT TRAN
PRINT 'Transakce proběhla bez problémů'
END TRY
BEGIN CATCH
ROLLBACK
PRINT 'Transakce byla odrolována zpět do konzistentního stavu'
END CATCH
```

Jelikož jsme se pokusili vložit do druhé tabulky (nizsi_uroven_skladu) nepovolenou hodnotu (404) která nemá v první tabulce odpovídající hodnotu, zobrazí se nám hlášení „***Transakce byla odrolována zpět do konzistentního stavu***“.

5.2.4. Triggers pro DML

Spouště (After triggers) bylo v předešlé verzi možné definovat pouze pro příkazy DML (Data Manipulation Language), jako jsou například INSERT, UPDATE nebo DELETE. Od verze 2005 můžeme tyto spouště definovat také pro příkazy DDL (Data Definition Language).

Například chceme-li zabránit vymazání tabulky z databáze, vytvoříme následující Trigger:

CREATE TRIGGER vymazanitabulky

ON DATABASE

FOR DROP TABLE

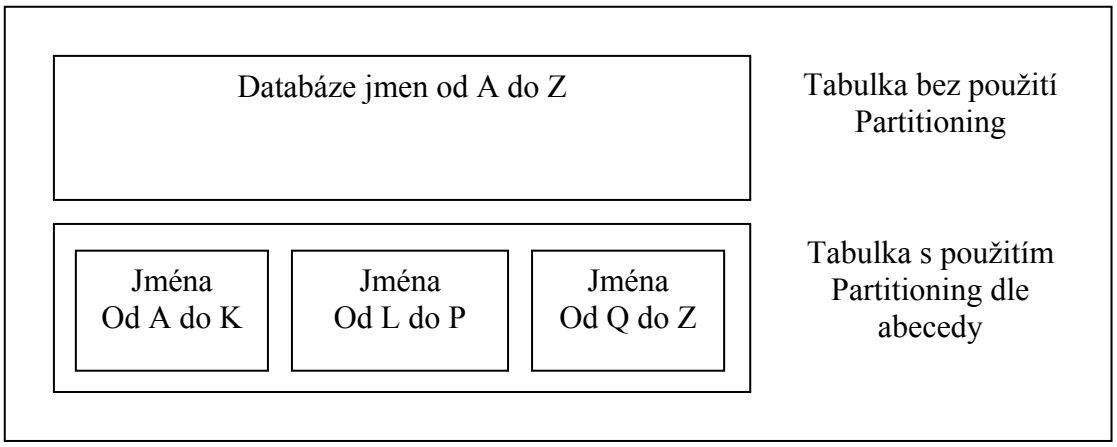
Budeme-li nyní chtít smazat tabulku, spoušť nám v tom zabrání. Dostaneme hlášení o spoušti, která toto způsobila. Trváme-li i přesto na smazání tabulky, musíme nejdříve zakázat samotný trigger a následně opakovat smazání tabulky.

Rozlišujeme triggery na BEFORE a AFTER, nebo-li spouště, které se vykonávají určitý programový kód před nebo po určité operaci.

Pomocí těchto spouští můžeme ovlivňovat integritu databáze, nebo můžeme zajistit automatické změny v databázi.

5.2.5. Partitioning

Pracujeme-li s databázemi velkého rozsahu, nebo s databázemi, které se s časem podstatně rozšiřují, je vhodné tabulku rozdělit na základě určité podmínky. Typicky dle abecedy, dle měsíců nebo roků. Jako příklad si můžeme uvést vydané faktury, které se mohou ukládat do menších tabulek dle měsíců, popřípadě dle roků nebo dle poboček v jednotlivých oblastech. Tuto funkci ovšem podporuje pouze verze SQL Sever 2005 Enterprise Edition a doporučuje se použití pro databáze velkého rozsahu a cca 8 procesorů a více.



Obrázek 9 – Partitioning[14)]

5.2.6. Nový datový typ XML

Jazyk XLM (Extensible Markup Language – Rozšiřitelný značkovací jazyk) se stále více používá pro přenos údajů mezi různými aplikacemi a pro publikování dokumentů.

Jazyk samotný nám popisuje strukturu dokumentu a pro vzhled se definuje připojený styl. Níže si uvedeme jednu z možností práce s XML.

Příklad pro **vytvoření XML dokumentu** naplněného pomocí stávajících dat z databáze AdwentureWorks (Standardní testovací databáze SQL Serveru 2005, k dispozici zdarma).

```
USE ADWENTUREWORKS
SELECT TOP(5) ProductID, Name, ProductNumber From Production.Product
ORDER BY ProductID
```

Tímto jsme si zobrazili prvních pět záznamů (řazeno dle sloupce ProductID) z tabulky ProductionProduct a to pouze sloupce ProductID, Name a ProductNumber. Následně z těchto dat vytvoříme soubor s daty XML.

```
DECLARE @x XML
SET @x = (SELECT ProductID as IDProduktu, Name as Jméno,
ProductNumber as ČísloProduktu
FROM Production.Product As Produkt
WHERE ProductID Between 1 and 2
FOR XML AUTO, ELEMENTS)
SELECT @x
```

Deklarujeme si proměnou „x“ typu XML. Následně proměnnou „x“ naplníme hodnotami ze sloupců ProductID, Name a ProductNumber z tabulky Production.Product, kde hodnota ProductID je mezi hodnotou 1 a 2. Příkaz Elements nastavený na hodnotu AUTO říká, že údaje budou vkládány jako elementy (nikoliv jako atributy).

Níže můžete vidět výsledek. Nejdříve pět prvních výpisů z tabulky Production.Product a dále již obsah samotného XML dokumentu.

ProductID	Name	ProductNumber
1	Adjustable Race	AR-5381
2	Bearing Ball	BA-8327
3	BB Ball Bearing	BE-2349
4	Headset Ball Bearings	BE-2908
316	Blade	BL-2036

tabulka 16 - XML – Zadání příkladu

Obsah samotného dokumentu XML:

```
<Produkt>
<IDProduktu>1</IDProduktu>
```

<Jméno>Adjustable Race</Jméno>

<ČísloProduktu>AR-5381</ČísloProduktu>

</Produkt>

<Produkt>

<IDProduktu>2</IDProduktu>

<Jméno>Bearing Ball</Jméno>

<ČísloProduktu>BA-8327</ČísloProduktu>

</Produkt>

Dále si na jednoduchém příkladu ukážeme **vytvoření zkušební tabulky se sloupcem XML**.

Vytvoříme si zkušební tabulku Vyrobky, kde MaterialID je primární klíč a sloupec Material typu XML:

CREATE TABLE Vyrobky (

MaterialID INT IDENTITY PRIMARY KEY,

Material XML NOT NULL)

A následně vložíme data do tabulky:

INSERT INTO Vyrobky VALUES

(' <Material>

<MaterialID>1</MaterialID>

<Jmeno>Sroubek</Jmeno>

<Popis>Nerez</Popis>

<Pocet>1570</Pocet>

<Cena>2,40</Cena>

</Material> ')

Tímto jsme vložili XML data do sloupce Materiál.

Jak i nad jinými sloupci, i nad XML lze vytvořit index pro rychlejší vyhledávání.

Standardní dotazovací jazyk nebyl vytvořen pro dotazování v datových typech XML a proto SQL Server 2005 obsahuje dotazovací jazyk XQUERY. Mezi jeho příkazy můžeme zařadit:

- **xml.exists:** jak z názvu vyplývá, dostaneme odpověď na otázku, zda daná hodnota ve sloupci existuje, či nikoliv,
- **xml.value:** vrací hodnotu daného prvku, určeného indexem v hranatých závorkách.

6. Srovnávací výkonový test Microsoft SQL server verze 2000 a 2005

Pokud provozovatel nebo správce používá starší verzi MS SQL 2000 a neuvažuje o zavedení nové verze například z důvodu některých novinek, které jsou uvedeny výše (datový typ XML, provázanost s Visual Studiem 2005, Snapshot, Partitioning, novinky v T-SQL atd.), bude se pravděpodobně ptát hlavně po výkonu a času zpracování jednotlivých dotazů.

Na následujícím příkladu výkonového testu a srovnání rychlostí zpracování transakcí se pokusím zodpovědět otázku přechodu na novou verzi pouze z důvodu rychlosti.

6.1. Základní údaje pro testování

Pro otestování rychlosti obou serverů jsem si vybral testovací nástroj Benchmark Factory for Database (dále BMF) společnosti Quest Software. Použil jsem omezenou trial verzi tohoto produktu (omezeno na maximálně 20 současně připojených testovacích uživatelů), která je poskytována zdarma.

BMF umožňuje vytvořit různé testovací algoritmy, různé testy a s některými se následně seznámíme. Samotný nástroj se dělí na řídící konzolu, která umožní definování způsobu provádění daného testu (nastavení transakcí, počty virtuálních uživatelů), dále se zde ukládají a zpracovávají samotné naměřené hodnoty a druhou část můžeme nazvat agenty, kteří generují požadovanou zátěž a zasílají získané výsledky do řídící konzole k vyhodnocení.

Stejný software pro testování databází použili také redaktoři odborného časopisu Connect ve svém článku. Dovolil jsem si tedy uvést také výsledky z jejich testování a pokusil se o srovnání mnou naměřených dat a jejich výsledků.

Test redaktorů časopisu Connect[15)] proběhl za výrazně lepších testovacích podmínek (jako základ testu posloužili server Abacus a-Server 1640T se dvěma procesory Opteron 275, 2 GB RAM, diskovým polem RAID 0, agenti pro generování zátěže byli nainstalováni na třech samostatných PC a pro monitorování a ovládání testovacího software byl použit další vyhrazený počítač). Redaktoři při testu použili komerční verzi softwaru BMF a mohli tak nasimulovat daleko větší množství připojených testovacích agentů.

V mém případě bylo použito pro celý test i monitoring pouze jednoho standardního PC a trial verze softwaru BMF. K testování jsem použil stejně jako redaktoři časopisu Connect standardní testy pro testování výkonu relačních databází:

- **AS3AP:** otevřený standard dle ANSI, srovnávací test (Benchmark) pro relační databáze. Obsahuje dvě části, pro jednoho uživatele (**Single-User**), který testuje přístupové metody a základní dotazovací optimalizaci (výstupní dotazy, Selections - výběry, Joints - spojení, Agregační a Up-date funkce), druhá část je **Multi-user** (Mnoho-uživatelská), která testuje rozdílné zatížení. Výsledky se udávají v transakcích za sekundu (víceuživatelské testy), popřípadě jako v čase potřebným k dokončení transakce (jednouživatelský test).
- **TPC-C:** standardní srovnávací test, který simuluje prostředí OLTP aplikace. Můžeme zde nalézt devět různých vzájemně propojených tabulek a pět transakcí, které běží souběžně z více terminálů. Každá transakce má definovanou – procentuální četnost spouštění, časy čekání pro vkládání dat a vyhodnocování získaných údajů. Výsledky jsou přepočítány na počet dokončených transakcí za minutu.

6.2. Test

Pro testování jsem měl k dispozici nepoměrně slabší vybavení, a i proto výsledky nebudou tolik průkazné, jako ve druhém případě. Rozdíl je jak v softwarovém tak hardwarovém vybavení. Nicméně můžeme prověřit výkon serveru při nižším zatížení.

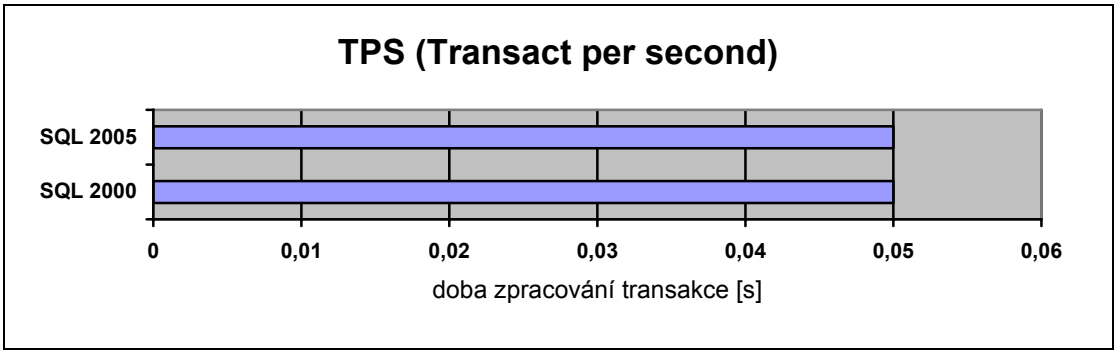
Byla použita Trial verze softwaru BMF omezená na 20 uživatelů.

Co se týče hardware, bylo použito jedno PC, které mělo na starosti jak monitorování, tak generování zátěže s následujícími parametry: procesor AMD 64 3000+, 1GB RAM s Windows XP SP2 a SQL Server 2000 MSDE, respektive SQL Server 2005 Express.

Zobrazíme si výkonové srovnání jednotlivých verzí a charakteristiky při zatížení serveru.

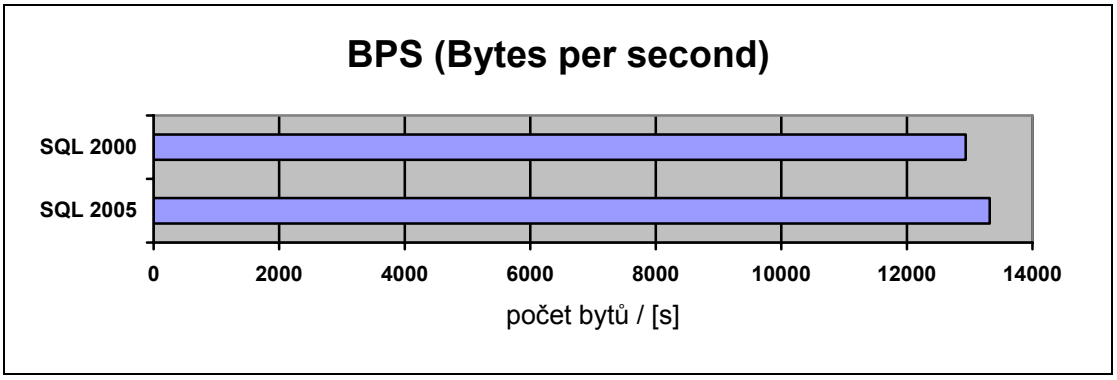
6.2.1. Test AS3AP

Výsledky **Single-User** testu, který nám udává dobu potřebnou na zpracování jedné transakce:



Graf 1 - AS3AP - TPS (Transact per second) – Single test

Na druhém grafu pak vidíme počet bytů zpracovaných serverem za sekundu:

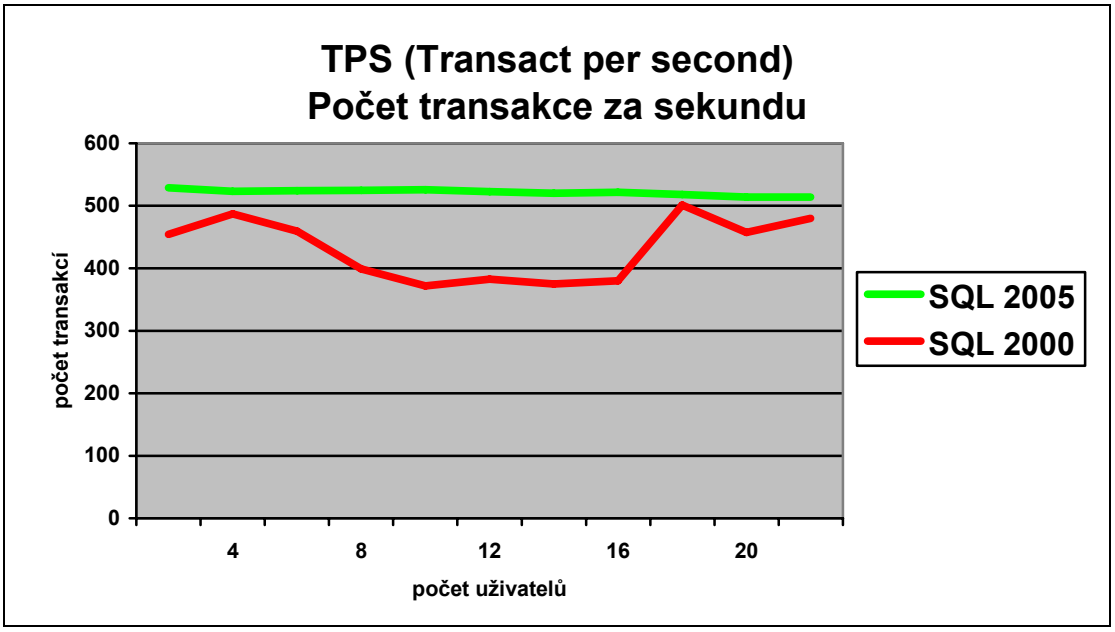


Graf 2 - AS3AP - BPS (Bytes per second) – Single test

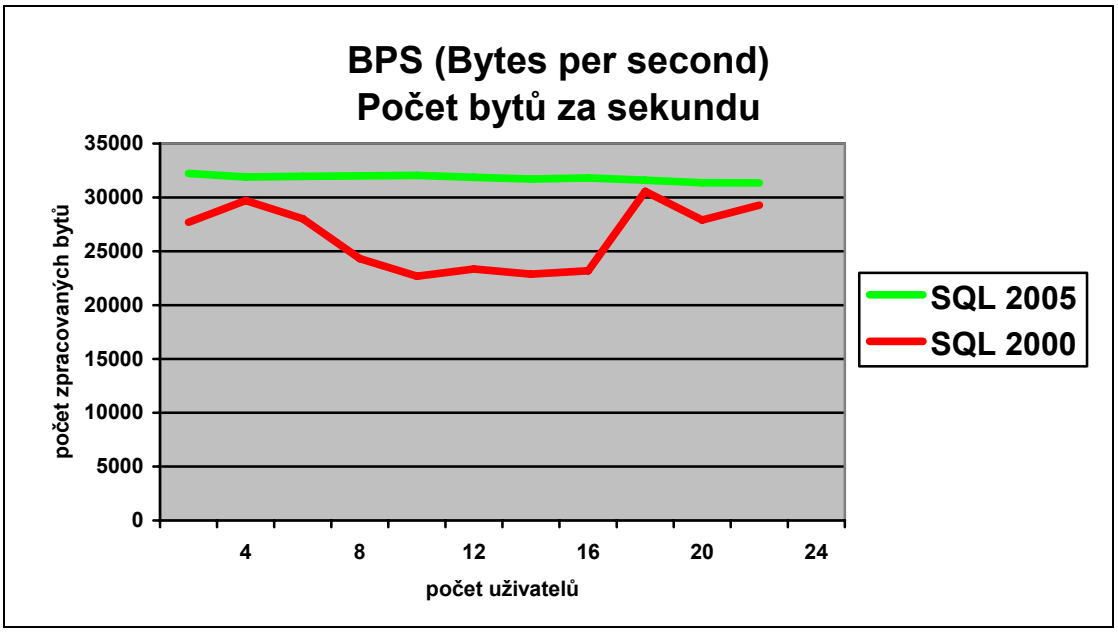
Jak můžeme vidět na prvním grafu, není mezi oběma verzemi v rychlosti provedení transakce prakticky žádný rozdíl. Na druhém grafu se již rozdíly obou serverů ukážou, nicméně jsou dosti nepatrné.

Z uvedených hodnot usuzuji, že nebude velkých rozdílů v rychlosti zpracování transakce mezi jednotlivými verzemi SQL Serveru přicházejících od jednoho klienta.

Jako další si ukážeme **Multi-User test** (stejným způsobem prováděný test jako v předchozím případě, zátěž je ale generována současně několika uživateli). Nejprve, jako při Single testu, si zobrazíme výsledky pro počet transakcí za sekundu a následně počet zpracovaných bytů za sekundu.



Graf 3 - AS3AP - TPS (Transact per second) - Multi User test

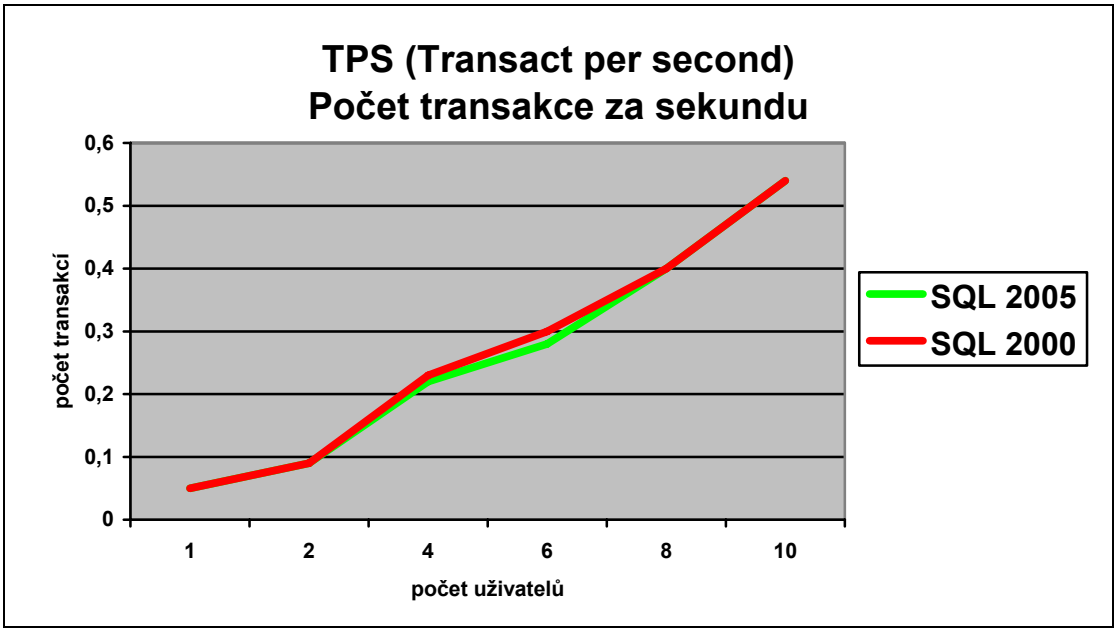


Graf 4 - AS3AP - BPS (Bytes per second) - Multi User test

Z této části testu vyšel jednoznačně vítězně SQL Server 2005 a to jak při porovnání zpracování transakcí za sekundu, tak i při zpracování bytů za sekundu. Z grafu je také jasně vidět větší stabilita výkonu u SQL Serveru 2005, u kterého nedochází k tak velkým výkyvů zpracování dat při postupném zvyšování uživatelů.

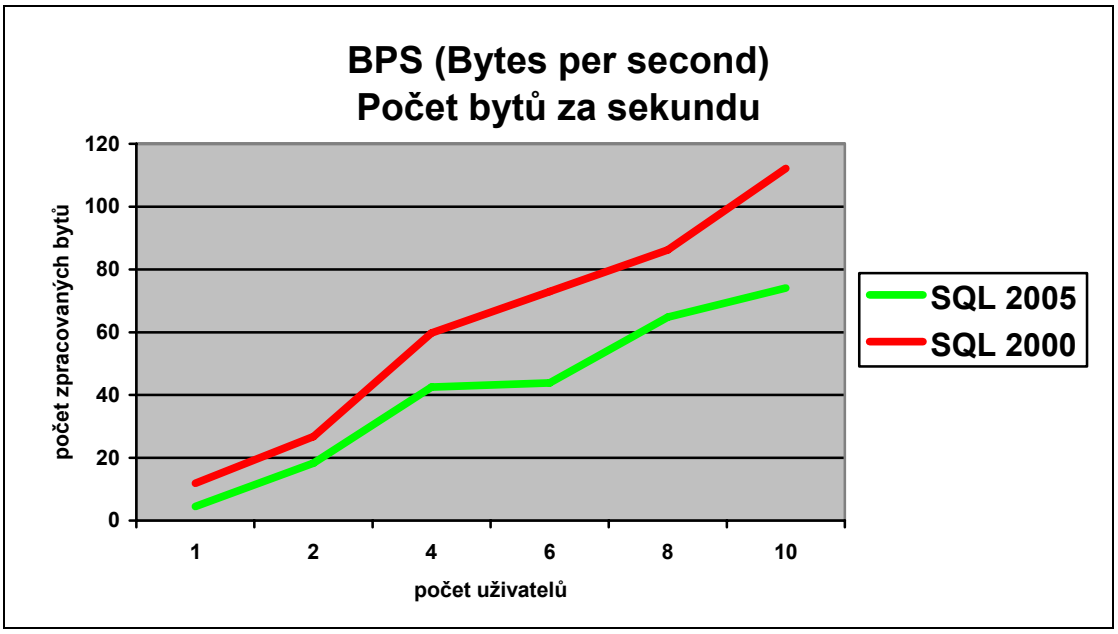
6.2.2. Test TPC-C

Opět si zobrazíme nejdříve výsledky testu pro zpracování transakcí za sekundu, tentokráte jako výsledek testu TPC-C.



Graf 5 - TPC-C - TPS - Transact per second

Oba výsledky se téměř dokonale shodují, nenalezl jsem větší rozdíl mezi oběma verzemi.



Graf 6 - TPC-C - BPS - Bytes per second

U tohoto testování došlo k mírnému překvapení, kdy novější verze SQL Serveru 2005 zpracovala viditelně méně bajtů při stejném počtu uživatelů.

6.3. Testovaní redaktorů časopisu Connect

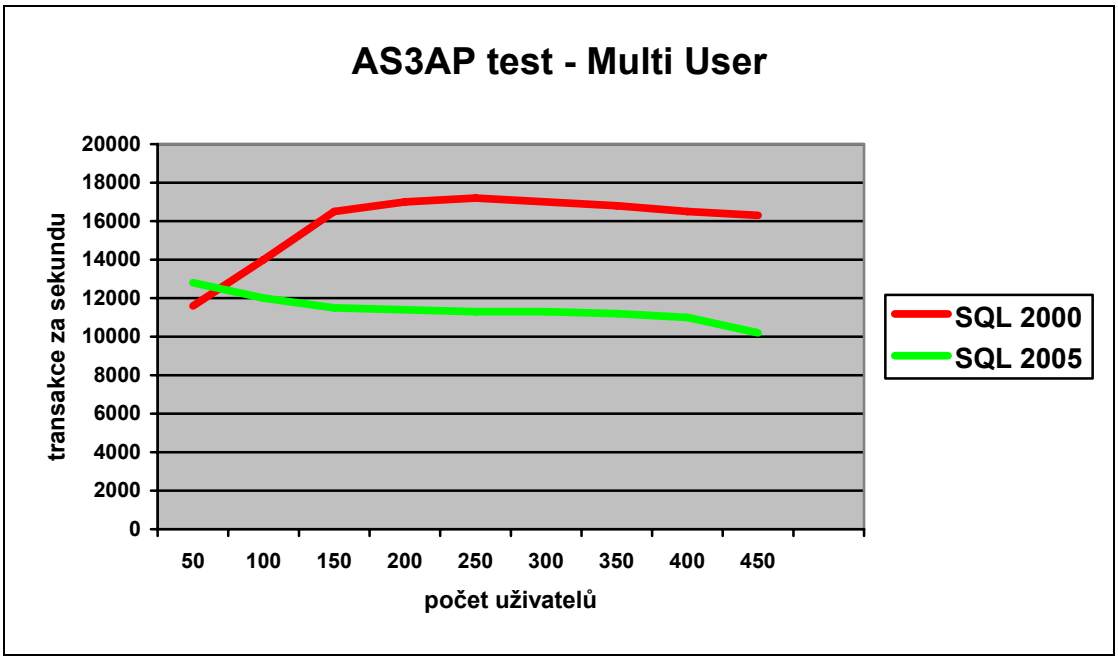
Nepoměrně lepší vybavení jak softwarové tak hardwarové, nám pomůže zobrazit více průkazné zátěžové testy.

Zátěžoví agenti byli rozmístěni na třech samostatných stanicích, řídící konzola BMF a Windows Spotlight na vyhrazené stanici. Vše propojeno pomocí LAN přes vyhrazený SWITCH. Testy byly prováděny na následujícím hardware a software: server Abacus a-Server 1640T se dvěma procesory Opteron 275 (dvou-jádrové 2,2 GHz), 2 GB RAM, operační systém Windows Server 2003 Standard x64 Edition, Servis Pack 1 a testovaný software SQL Server 2000 a 2005 Standard Edition. Velikost vygenerované databáze, pomocí níže uvedeného nástroje, byla 20 GB.

6.3.1. Test AS3AP

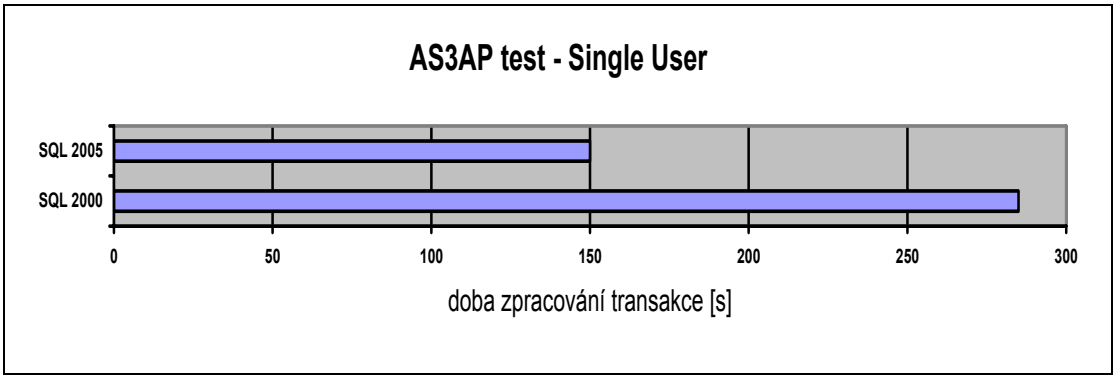
Nejprve si zobrazíme Multi-User test. Jak můžeme vidět na následujícím obrázku, poražena odešla novější verze SQL Server 2005 a to hlavně při větším zatížení pomocí většího počtu uživatelů (a to přibližně od 75 uživatelů). Ze systémových statistik bychom mohli dohledat, že důvodem může být lepší využití procesorů u verze 2000. Při nižším počtu uživatelů (do cca 75) SQL Server 2005 prokazuje rychlejší reakci,

která může být způsobena například nižším výskytem dlouhých časů u jednotlivých transakcí.



Graf 7 - AS3AP test - Multi-User[15)]

Následuje test pro Single-User:



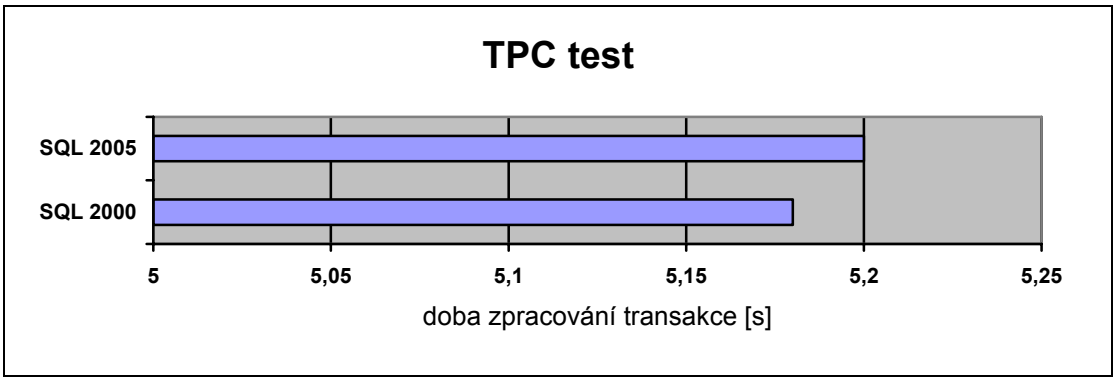
Graf 8 - AS3AP test - Single User[15)]

U tohoto testu můžeme vidět jasnou převahu verze SQL 2005, která potřebuje na zpracování jedné transakce podstatně menší časový úsek, nežli verze SQL 2000.

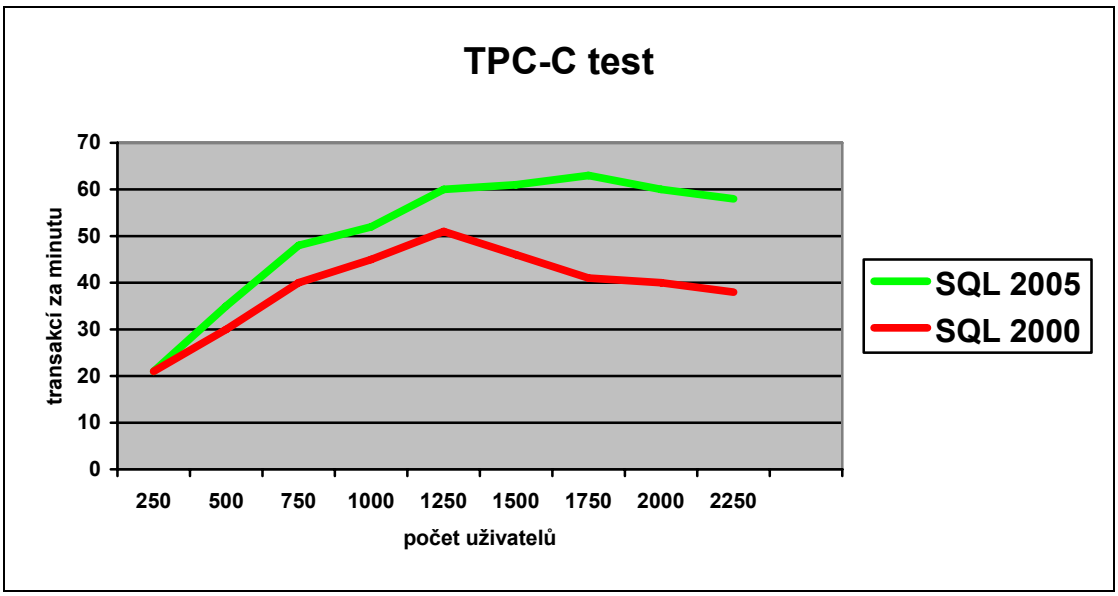
6.3.2. Test TPC-C

U tohoto testu byly použity dvě varianty. Při prvním testu, který prověřoval funkčnost testovacího zapojení a databázových serverů, oba servery dosáhly téměř stejného výsledku při zpracování transakce. Při druhém testu se hledal maximální počet uživatelů, kteří mohou souběžně pracovat na jednom serveru. Microsoft podstatně zapracoval na stabilitě SQL Serveru 2005 oproti předchozí verzi, například se již nevyskytují v takové míře Dead-Locks (neboli „zamrznutí“, kdy jedna transakce čeká

na dokončení druhé, která naopak čeká na dokončení první transakce. Toto zaseknutí většinou server řeší odvoláním jedné transakce po určitém Timeout – vyprší čas určený pro vyřízení transakce) a proto SQL Server 2005 v podstatné míře porazil svého předchůdce.



Graf 9 - TPC-C test první[15)]



Graf 10 - TPC-C test druhý[15)]

Jak je vidět z těchto prvních testů, SQL server 2005 je ve většině případů rychlejší i když se najdou výjimky, při kterých verze 2000 vítězí (náš první test AS3AP – Multi-User).

6.4. Závěr testování

Průkaznost druhého testování má jistě mnohem větší váhu ve srovnání s mnou naměřenými hodnotami, nicméně jsme na prvním příkladě mohli pozorovat zatížení serverů při menším počtu uživatelů a zpracování menšího objemu dat, které jistě využívá nemalá část serverů.

V mém prvním testu zvítězila jednou verze SQL Server 2000 a jednou verze SQL Server 2005. Každý z testů AS3AP a TPC-C dopadl jinak, jednou zvítězila verze 2000 a jednou verze 2005. Tuto okolnost můžeme přisoudit menšímu počtu uživatelů a menšímu počtu zpracovaných transakcí. Proto můžeme vyřknout výsledek z prvního testování nerozhodně. Opravdová zátěž se prokázala teprve ve druhém testování.

Druhý test nám teprve ukázal řádnou zátěž obou serverů, nicméně ani zde nemůžeme určit jasného vítěze. V testech AS3AP Single-User měla jasně navrch verze 2000, ovšem při použití Multi-User testu pro AS3AT zvítězila verze 2005. Stejně tak dopadl i test TPC-C, kdy při zpracování jedné transakce zvítězila verze SQL 2000, ovšem zpracování transakcí při zatížení více uživateli verze 2005 jasně převýšila verzi 2000.

Nemáme tedy jasného vítěze a nelze ani jasně doporučit přechod na novou verzi pouze z důvodu rychlosti. Nicméně novinky v ostatních oblastech, ať už v oblasti T-SQL, uživatelského prostředí nebo stability a podobně, jistě mnoho provozovatelů přiměje ke koupi nové verze.

S koncem roku 2007 ovšem přichází otázka, zda raději nepočkat na začátek dalšího roku a na novou verzi MS SQL 2008, která je již dostupná v demoverzi.

7. SQL Server 2008

Microsoft plánuje vydání trojice produktů SQL Server 2008, Windows 2008 Server a Visual Studio 2008 na počátek roku 2008, pravděpodobně 27.02.2008. Nyní je k dispozici demoverze SQL Server 2008 JULY CPT. Uvedeme si některé důležité novinky v přicházející verzi:

- v rámci bezpečnosti verze 2008 přináší mnohé vylepšení. Například vytváření šifrované zálohy, popřípadě umožňuje nastavit počet povolení obnovy z dané zálohy, ale i ukládání klíčů na speciální hardware.
- Přidání procesoru, bez nutnosti vypnutí nebo restartu.
- Již není omezena velikost uživatelsky definovaných datových typů.
- Setkáme se s novými datovými typy – Date, Time, DateTime Offset (údaje s časovou zónou).
- Dosud neuvedené datové typy pro GPS (Location a Geometry).
- Datový typ File Stream – ukládá nestrukturovaná data.
- SSIS (Server Integration Services) podporuje multiprocesorové zpracování (dříve max na dvou procesorech).

8. Závěr

Po výčtu všech novinek a vylepšení, které byly uvedeny výše, by patrně většina z nás volila přechod z verze SQL 2000 na verzi SQL 2005. Nelze to ovšem říci v každém případě. Někteří správci mohou být spokojeni s verzí SQL 2000, novinky týkající se například propojení s Visual Studio 2005, XML datový typ, nebo jiné novinky prostě nevyužijí a budou se ptát pouze po výkonu. V tomto případě, jak jsme si ukázali na dvou případech, nelze jednoznačně doporučit přechod na novou verzi. Jak ukázal náš test, jestliže testujeme pouze výkon a čas zpracování, není jednoznačného vítěze a oba servery mají prakticky stejný výkon.

Starší verze SQL 2000 je samozřejmě pořád velmi rozšířena, ovšem její podpora pro nynější operační systém Windows Vista a nový Windows Server 2008 bude pouze minimální a to hlavně z důvodu zabezpečení.

V letošním roce proto přichází otázka, máme-li verzi 2000, nemáme-li raději vyčkat na vydání SQL Serveru 2008, který má vstoupit do obchodů začátkem roku 2008. Jestli budou vylepšení alespoň ve stejném rozsahu jako z verze 2000 na verzi 2005, určitě se máme se na co těšit.

9. Seznam tabulek

tabulka 1 - Nultá normální forma 13
tabulka 2 - První normální forma – zadání 13
tabulka 3 - První normální forma - výsledná tabulka 1/2 13
tabulka 4 - První normální forma - výsledná tabulka 2/2 14
tabulka 5 - Druhá normální forma – zadání 14
tabulka 6 - Druhá normální forma – výsledná tabulka 1/2 14
tabulka 7 - Druhá normální forma – výsledná tabulka 2/2 15
tabulka 8 - Třetí normální forma – zadání 15
tabulka 9 - Třetí normální forma – výsledná tabulka 1/3 15
tabulka 10 - Třetí normální forma – výsledná tabulka 2/3 16
tabulka 11 - Třetí normální forma – výsledná tabulka 3/3 16
tabulka 12 - TOP n – zadání 38
tabulka 13 - TOP n – výsledek 39
tabulka 14 - RANK, DENSE RANK, ROW_NUMBER, NTILE – zadání 39
tabulka 15 - RANK, DENSE RANK, ROW_NUMBER, NTILE – výsledek 40
tabulka 16 - XML – Zadání příkladu 43

10. Seznam obrázků

Obrázek 1 - Připojení k serveru MS SQL 2005 pomocí ODBC[1)] 30
Obrázek 2 - Připojení k serveru MS SQL 2005 pomocí OLE DB[1)] 31
Obrázek 3 - OLE DB a ODBC - Přístup aplikace pomocí různých rozhraní 32
Obrázek 4 - Připojení k serveru MS SQL 2005 pomocí ADO.NET[1)] 32
Obrázek 5 - Připojení k serveru MS SQL 2005 pomocí SQL Server Native Client[1)] 33
Obrázek 6 - .NET Framework 33
Obrázek 7 - Struktura SQL 2005[16)] 34
Obrázek 8 - SQL Server Notification Services (SSNS) 36
Obrázek 9 – Partitioning[14)] 42

11. Seznam grafů

Graf 1 - AS3AP - TPS (Transact per second) – Single test 46
Graf 2 - AS3AP - BPS (Bytes per second) – Single test 47
Graf 3 - AS3AP - TPS (Transact per second) - Multi User test 47
Graf 4 - AS3AP - BPS (Bytes per second) - Multi User test 48
Graf 5 - TPC-C - TPS - Transact per second 48
Graf 6 - TPC-C - BPS - Bytes per second 49
Graf 7 - AS3AP test - Multi-User[15)] 50
Graf 8 - AS3AP test - Single User[15)] 50
Graf 9 - TPC-C test první[15)] 51
Graf 10 - TPC-C test druhý[15)] 51

12. Literatura

1. Solid Quality learning. *Microsoft SQL Server 2005: základy databází. Krok za krokem.* Brno: Computer Press, 2007. 320 s. ISBN 978-80-251-1524-4
2. VIEIRA, Robert. *SQL Server 2000 Programujeme profesionálně*. Praha: Computer Press, 2001. 1206 s. ISBN: 8072265067.
3. *MS SQL Server 2000 a 2005 Books Online*. URL: < http://www.microsoft.com/downloads/details.aspx?FamilyID=BE6A2C5D-00DF-4220-B133-29C1E0B6585F&displaylang=en > [cit. květen 2007]
4. CODD, E.F.: The relation model for database management: version 2. IBM Research Lab, San Jose, CA.1990. ISBN: 0-201-14192-2 . Dostupné na: http://portal.acm.org/citation.cfm?id=77708&coll=GUIDE&dl=GUIDE&CFID=27572055&CFTOKEN=11527749
5. www.dbsvet.cz: InterSystems lepší IBM? [online]. 2007 [cit. 2007-01-02]. Dostupný na: http://www.dbsvet.cz/view.php?cisloclanku=2007020102
6. www. Sybase.cz: Adaptive Server Enterprise. [online]. 2007 [cit. 2007-04-02]. Dostupný na: http://www.sybase.cz/buxus/generate_page.php?page_id=112&view=1&typ_p=287 nebo http://www.sybase.com/products/databasemanagement/adaptiveserverenterprise
7. JAKUBČÍK, Ondřej: Srovnání databázových serverů. [online]. 2007 [cit. 2007-03-06]. Dostupný na http://www.linuxexpres.cz/software/srovnani-databazovych-serveru
8. www.wikipedia.cz: Oracle. 2007 [cit. 2007-01-08]. Dostupný na http://cs.wikipedia.org/wiki/Oracle_Corporation
9. CODD, E.F.: A relational model of data for large shared databank. IBM Research Lab, San Jose, CA.1983. ISSN:0001-0782 . Dostupné na: http://portal.acm.org/citation.cfm?id=358007
10. Firemní materiály Oracle Česká republika. Dostupný na http://www.oracle.com/global/cz/database/edice.html
11. Firemní materiály Intersystems corporations. Dostupné na http://www.intersystems.com/
12. http://www.w3.org/
13. http://www.mysql.com/
14. www.dbsvet.cz, MS SQL 2005, 21.12.2005 URL: http://www.dbsvet.cz/view.php?cisloclanku=2006080101 [cit. květen 2007]

15. HONEK, Lukáš. *Je SQL 2005 rychlejší než SQL 2000?* [online]. 2006. [cit. 5. 4. 2006]. Dostupný z: http://connect.zive.cz/index.php?q=node/227
16. *Přehled produktu SQL Server 2005.* [online]. 2005. [cit. 7. 11. 2005]. Dostupný z: http://www.microsoft.com/cze/windowsserversystem/sql/prodinfo/overview/default.mspx
17. *Microsoft oznámil ceny a podrobné informace pro SQL Server 2005*. [online]. 2005. [cit. 3. 3. 2005]. Dostupný z: http://www.microsoft.com/cze/presspass/msg/20050303_news1.mspx
18. DB2 Product Family. Firemní materiály IBM. Dostupné na: www.ibm.com/db2
19. www.fi.muni.cz: Relační vs. objektově-relační vs. objektové databáze. [online]. Dostupný na http://www.fi.muni.cz/~xbatko/oracle/compare.html
20. www.connect.cz: InterSystems vydalo Caché 2007 [online]. 2006 [cit. 2006-13-11]. Dostupný z WWW <http://connect.zive.cz/?q=node/449>.

ÚDAJE PRO KNIHOVNICKOU DATABÁZI

Název práce	Microsoft SQL Server 2005
Autor práce	Preisler Martin
Obor	Systémové inženýrství a informatika
Rok obhajoby	2007
Vedoucí práce	Mgr. Karel Naiman
Anotace	Cílem bakalářské práce je seznámení se s relačními databázemi, zejména s MS SQL Serverem 2005, zvláště pak výkonové srovnání s předchozí verzí MS SQL 2000 a otázka přechodu na novější verzi. Dále jsou uvedeny novinky v T-SQL, struktura SQL 2005 a přístup k datům v nové verzi.
Klíčová slova	Microsoft SQL Server 2005, Microsoft SQL Server 2000, SQL, výkonové testy, T-SQL, relační databáze, normalizace, standardizace, ODBC, OLE DB, ADO.NET, Native client, .NET Framework.