Microsoft .net

Microsoft

Ľuboslav Lacko

Microsoft SQL Server 2000

Reportovacie služby

Čo je managed reporting?
Architektúra a filozofia produktu
Reportovacie služby z pohľadu vývojára
Reportovacie služby z pohľadu administrátora
Embedding do zákazníckych a partnerských riešení

Microsoft SQL Server 2000 Enterprise Edition

Obsah:

Linky na ktorých je možné získať trial verzie produktov

Visual Studio .NET 2003	http://msdn.microsoft.com/vstudio/productinfo/trial/default.aspx alebo dočasne tiež http://www.microsoft.cz/akce/VS.NETResKit/
SQL Server 2003	http://www.microsoft.com/sql/evaluation/trial/default.asp
SQL Server SP3	http://www.microsoft.com/sql/downloads/2000/sp3.asp
SQL Report Services	http://www.microsoft.com/sql/reporting/productinfo/trial.asp

Informácie ohľadne českej lokalizácie sú na adrese www.adastra.cz/reportingservices

Príklady k zborníku je možné stiahnuť na adrese http://msdn.microsoft.cz/docs/BrozuraSQLRS.zip

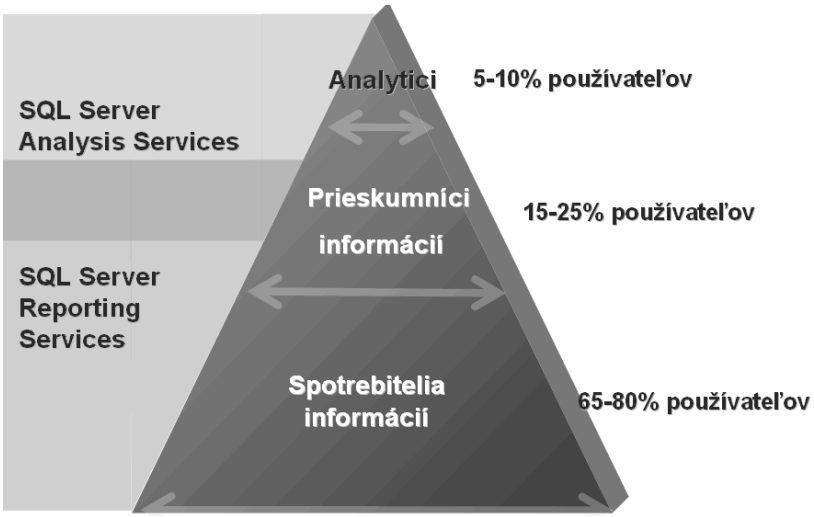
Úvod

Kapitola 1:

Čo je managed reporting?

Úvod

Intenzívne nasadzovaním informačných technológií do rozličných odvetví ľudskej činnosti prináša so sebou jeden zákonitý jav, ktorým je zhromažďovanie veľkého množstva rôznorodých údajov. Zhromažďujú sa údaje z technologických zariadení, firemnej administratívy, odbytu a podobne. Ukladanie údajov do databázy je totiž len jedna stránka problematiky, veľakrát sa robí najmä preto, že niekedy v budúcnosti by sa zhromaždené údaje mohli k niečomu hodiť. Údaje sa potom spracovávajú a výsledky spracovania sa publikujú na dvoch úrovniach.
Na krátkodobej operačnej úrovni a na dlhodobej strategickej úrovni. Ako vyplýva z obrázku 5 až 10 percent používateľov používa výsledky analýz, 15 až 25 percent používateľov tieto informácie skúma a hľadá v nich súvislosti. Najväčšia skupina používateľov informácie používa vo forme rôznych výpisov a reportov.



Obr. 1.1 Oblasť použitia a nasadenia služieb SQL Servera

Paradoxne už dávno existoval doplnok k SQL Serveru s názvom Analytické služby, ktorý sa distribuuje spolu s SQL Serverom na inštalačnom CD. Môžeme si ich pokojne vyskúšať aj v prípade ak máme 120 dňovú trial verziu SQL Servera No dosiaľ neboli k dispozícii reportovacie služby. Nie je to žiadna tragédia, pre publikovanie údajov na webe alebo na firemnej sieti sú k dispozícii rôzne metódy a nástroje. Môžeme napríklad vytvárať rôzne jednoúčelové alebo viac – menej univerzálne aplikácie. Situácia sa ale čiastočne zmení po komerčnom uvedení reportovacích služieb, kedy pre vytváranie reportov získame skutočne efektívne nástroje

Obrázok navodzuje dva prechody. Jednak od transakčných databáz k analytickým databázam a taktiež od reportovacích služieb k analytickým službám. Obidva prechody sú do určitej miery paralelné a taktiež do istej miery neostré. Oblasť použitia transakčných databázových systémov **TPS** (Transaction Processing Systems) je takmer neobmedzená. Umožňujú klientom databázového servera vykonávanie veľkého množstva on - line transakcií väčšinou v reálnom čase. Sú to databázy v bankách, skladoch, obchodných reťazcoch a podobne. Ak transakčný databázový systém pokrýva väčšinu podnikových aktivít nazývame ho niekdedy aj systém **ERP** (Enterprise Resource Planning). Charakteristickou črtou týchto systémov je skutočnosť, že k zdroju údajov v rovnakom čase pristupuje veľké množstvo používateľov, ktorí údaje z databázy čítajú, iní do nej zapisujú, prípadne niektorí vykonávajú aj jednoduchšie analýzy. Nie je to tlačová chyba. Aj nad **OLTP** (On Line Transactional Processing) databázami je možné vytvárať nielen reporty, ale je možné vybudovať aj **OLAP** (On line Analytical Processing) analytické aplikácie. Nie je to však najvýhodnejšie riešenie, pretože údaje v transakčných databázach sú usporiadané v normalizovaných tabuľkách, ktoré by mali vyhovovať podmienkam druhej, alebo tretej normálnej formy. To znamená veľa atomických, relačne zviazaných tabuliek. Analýza veľkého množstva takto uložených údajov by preto vo väčšine prípadov mohla byť neefektívna a značne pomalá.
Už spomínaným neostrým rozhraním medzi relačnými a transakčnými systémami sa dostávame k systémom pre podporu riadenia a rozhodovania. Hlavnou úlohou týchto systémov je poskytovanie kvalitných informácií pre riadiacich pracovníkov. Dostávame sa na firemnú „operatívnu taktickú úroveň“. Systémy **MIS** (Management Information System) už poskytujú riadiacim pracovníkom rôzne komplexné prehľady a zostavy, agregované podľa rôznych hľadísk, napríklad časových, geografických, organizačných a iných. Do systémov MIS vstupujú údaje z transakčných systémov. Systémy MIS však nevynikajú operatívnosťou ani pružnosťou. Životný cyklus MIS bol zhruba nasledovný. Požiadavky na zostavu sa odoslali vývojovému tímu MIS, ktorý vytvoril zostavu a poskytol ju manažérom

až po určitom čase, spravidla po niekoľkých dňoch, týždňoch alebo dokonca až po niekoľkých mesiacoch. Určitou nadstavbou systémov MIS sú systémy **DSS** (Decision Support Systems), ktoré sú už na rozhraní taktického a strategického rozhodovania. Poskytujú riadiacim pracovníkom napríklad výsledky pomerne zložitých analýz. Analytické zostavy a reporty sú parketou pre **OLAP** (On line Analytical Processing). Typické nasadenia týchto systémov analýza veľkého množstva údajov. Výsledkom analýzy sú súhrny a reporty, ktoré slúžia manažérom ako podklady pre ich rozhodnutia, či už v oblasti riadenia firmy, riadenia ekonomických a technologických procesov a podobne.

Snahou Microsoftu je vytvoriť jednu platformu pre široké spektrum štruktúrovaných údajov, či už sa jedná o relačné, hierarchické alebo multidimenzionálne údajové štruktúry. Taktiež je potrebné reporty integrovať do aplikácií, ktoré sa bežne používajú, či už Office, prípadne Share Point pri organizácii tímovej práce, návrhy reportov integrovať do používaných vývojových prostredí, v prípade Microsoftu do MS Visual Studia .NET 2003.

Managed reporting

Hlavnou výhodou „managed reportingu“ je skutočnosť, že metadáta (RDL súbory) na základe ktorých sa generujú reporty sú uložené centrálne a teda môžu byť aj centrálne spravované. To znamená, že môžeme reporty organizovať do rôznych zostáv, adresárov a podobne, sprístupňovať ich na rôznej úrovni používateľom a skupinám používateľov, napríklad pre obchodné oddelenie, personálny manažment, ekonomický a výkonný manažment a podobne. Prístup k reportom potom môžeme riadiť cez prístup k adresárovým štruktúram.

Stratégia a ciele reportovacích služieb

Reportovacie služby by mali slúžiť pre podporu rozhodovania na všetkých stupňoch organizačnej infraštruktúry. Návrhom reportu, nech by bol hocijako kvalitný a trebárs aj nadčasový a zohľadňoval by všetky požiadavky sa celý proces reportovania nekončí. V ďalšej kapitole bude zdôraznený životný cyklus reportu, kde okrem návrhu reportu vystupuje do popredia aj jeho správa a taktiež spôsob ako sa k reportom dostanú jeho adresáti.

Začnime cieľom. Cieľom by sme mohli nazvať hlavný dôvod pre ktorý reportovacie služby do firmy zavádzame. Podobne ako databázový server ani reportovacie služby sa nezavádzajú na klientské počítače (s výnimkou vývojárskych), ale nasadzujú sa v rámci podnikovej informačnej infraštruktúry, aby pomáhali zamestnancom na všetkých stupňoch efektívny prístup k údajom a tým ich podporili v ich činnosti, alebo v prípade manažérov v procese rozhodovania.

Ak by sme chceli reporty nejakým spôsobom kategorizovať, mohli by sme pre tento účel zvoliť viacero kritérií. Prvá kategorizácia, ktorá nás napadne bude rozdelenie reportov na
– tradičné
– interaktívne

Tradičné reporty v elektronickej podobe sa principiálne nijako nelíšia od „tradičných papierových“ reportov. Môžeme v nich hlavne čítať a listovať. Naproti tomu interaktívne reporty môžeme pomocou rôznych ovládacích prvkov prispôsobiť tak, aby sme jednak získali tie informácie, ktoré potrebujeme a v takom tvare, v akom ich práve potrebujeme, prípadne v akom tvare ich chceme prezentovať svojmu obchodnému partnerovi a podobne.

Ďalšie delenie by mohlo byť podľa oblasti a filozofie nasadenia.

– Enterprise
– Embedded
– B2B

Enterprise Reporting

Na tejto podnikovej úrovni je reportovanie nasadené formou „In-house“ reportov napríklad pre obchodné oddelenie, finančné oddelenie, oddelenie ľudských zdrojov, vo sfére CRM a podobne. Údaje sú buď v podnikových databázach alebo v dátových skladoch. Výhodou je, že údaje sú už predspracované a pretransformované v etape ETL, a prenesené z produkčných systémov do dátových skladov (data warehouse), prípadne dátových trhov (data mart) Reporty z reportovacích služieb potom vhodne dopĺňajú údaje z analytických business intelligence aplikácií. Nasadenie reportovacích služieb je v tomto prípade prevažne na úrovni podnikových portálov, takže koncoví používatelia k nim pristupujú v rámci podnikového Intranetu.

Embedded Reporting
Na tejto úrovni sú reporty integrálnou súčasťou aplikácií, čo prináša rozšíriteľnú a škálovateľnú architektúru informačného systému

B2B Reporting
Na úrovni B2B opúšťame relatívne bezpečnú pôdu svojej firmy alebo organizácie a časť údajov poskytneme partnerom. Schválne pripomenieme jednu vetu z úvodu state „aby sme získali tie informácie, ktoré potrebujeme a v takom tvare, v akom ich práve potrebujeme, prípadne v akom tvare ich chceme prezentovať svojmu obchodnému partnerovi“. Totiž nemá zmysel upravovať skutočnosť pokiaľ sa jedná o podklady pre vlastné rozhodovanie, prípadne pre rozhodovanie manažérov vlastnej firmy. Maximálne môžeme "zakryť" určité informácie, ktoré ten ktorý pracovník pre svoje rozhodovanie nepotrebuje. Ak však prezentujeme údaje svojim obchodným partnerom, nevyzradíme im pochopiteľne slabé miesta svojej firmy, prípadne im neposkytneme údaje, na základe ktorých by títo mohli odhaliť naše potenciálne slabiny.

Licencovanie reportovacích služieb
Aj napriek tomu, že tento zborník má za úlohu predstaviť reportovacie služby z technického hľadiska, je potrebné pár slovami spomenúť aj licenčné podmienky. Reportovacie služby sú licencované rovnako ako SQL Server 2000. To znamená, že pre majiteľov licenčných práv na SQL Server 2000 sú reportovacie služby vlastne add – on balíkom zdarma. Týka sa to len SQL Serveru ako takého .Nie je možné napríklad používať reportovacie služby pre MSDE. Doterajší zákazníci s platnou licenciou na SQL Server teda za reportovacie služby neplatia nič, za predpokladu, že si reportovacie služby nainštalujú na tento server. Ak si však reportovacie služby nainštalujú na oddelený server, tak musia platiť aj serverovú licenciu pre tento SQL Server

Inštalácia produktu
Reportovacie služby SQL servera sú k dispozícii v štyroch verziách

- Standard Edition
- Enterprise Edition
- Developer Edition
- Evaluation Edition

Standard Edition
Táto verzia je určená pre produkčný server, v jedno počítačovej konfigurácii. Pomocou verzie Standard Edition môžeme vytvoriť centralizované riešenie, kde sú nainštalované zároveň reportovacie služby, databáza aj príslušné nástroje. Podmienkou pri inštalácii je, aby reportovacie služby, ich databáza (systémová, ktorú tieto služby využívajú pre ukladanie parametrov a pracovných údajov, nie databáza z ktorej údajov chceme reporty vytvárať) a Report Manager musia byť nainštalované na tom istom počítači. Táto verzia sa hodí pre malé firmy, prípadne malé pracovné skupiny, kde nepredpokladáme masívny a intenzívny prístup k reportom.

Enterprise Edition
Verzia Enterprise Edition je primárne určená pre produkčné servery do podnikového prostredia a pre veľké organizácie, kde sú požiadavky na reporty veľmi časté. Táto verzia je najkomplexnejšia a poskytuje vysoký výkon a škálovateľnosť. Verzia podporuje data-driven subscriptions, kedy sú reporty generované na základe zmien v produkčnej databáze.

Developer Edition
Vývojárska verzia slúži pre podporu vývoja aplikácií. Verzia je prakticky totožná s Enterprise Edition, no licencia platí len pre vývoj a testovanie aplikácií. Túto verziu nie je možné nasadiť na produkčný server.

Evaluation Edition
Evaluation Edition je totožná so Standard Edition a teda je určená pre jednopočítačovú konfiguráciu. Je možné ju používať 120 dní.

Komponenty pre inštaláciu

Komponenta	Typ počítača
Report server	Server s nainštalovaným SQL Serverom a Internet Information Services (IIS)
Report Manager	Web server
Databáza report servera	Server s nainštalovaným SQL Serverom
Administrátorské nástroje a utility	Klientská stanica s prístupom na report server
Report Designer	Klientská stanica s nainštalovaným vývojovým prostredím Microsoft Visual Studio .NET 2003
Nápoveda – Reporting Services Books Online	Klientská stanica
Príklady	Klientská stanica

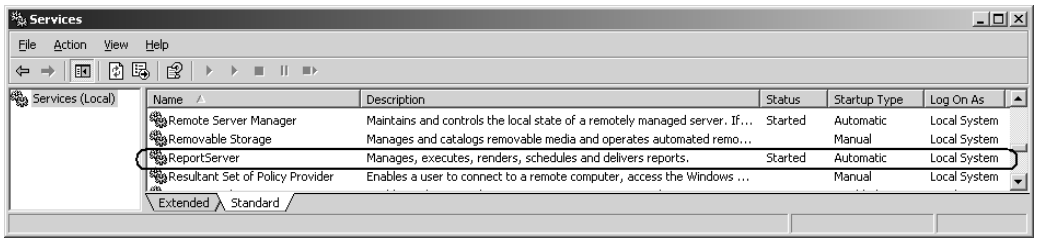
Ak by sme mali odporučiť kombináciu pre prvé pokusy a štúdium reportovacích služieb, bol by to počítač s nainštalovaným operačným systémom Windows XP a vyšším (Windows Server 2003), nainštalovaným a nakonfigurovaným web serverom IIS, databázovým serverom Microsoft SQL Server 2000 s nainštalovaným SP 3, a vývojové prostredie Microsoft Visual Studio .NET 2003. Pri inštalácii Visual Studia sa automaticky nainštaluje aj technologická platforma .NET Framework 1.1.

Zaujímavou variantou je vytvoriť si pre tieto účely virtuálny počítač (Connectix, Virtual PC 2004) a spomínané programy nainštalovať naň

Nakoľko v procese inštalácie budeme musieť zadať, ktoré komponenty chceme nainštalovať, nezaškodí sa s nimi pred samotnou inštaláciou zbežne zoznámiť, aby sme vedeli, čo nainštalovať a čo naopak potrebovať nebudeme. Reportovacie služby obsahujú tieto komponenty

Report Server

Report server je bezstavový server spravujúci metadáta (údaje o údajoch), definície objektov a podobne. Tieto údaje má vo vlastnej databáze, ktorá je pod správou SQL Servera 2000. Je implementovaný ako služba operačného systému Windows, rovnako ako webový server. Presvedčíme sa o tom (samozrejme až po nainštalovaní produktu), ak v administrátorských nástrojoch operačného systému spustíme utilitu Services



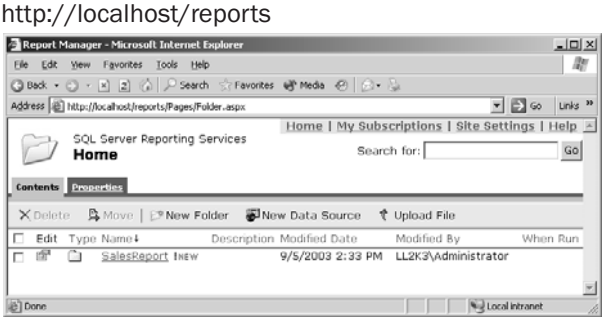
Obr. 1.2 Report server ako služba operačného systému

Report server je možné ovládať cez web – nachádza sa vo virtuálnom adresári a je prístupný cez URL adresu http://nazov_pocitaca/ReportServer ktorú môžeme prípadne modifikovať pri inštalácii

Report Manager

Nástroj Report Manager sa používa pre správu reportov. Nachádza sa vo virtuálnom adresári a je prístupný cez URL adresu, ktorú môžeme zadať pri inštalácii

http://localhost/reports

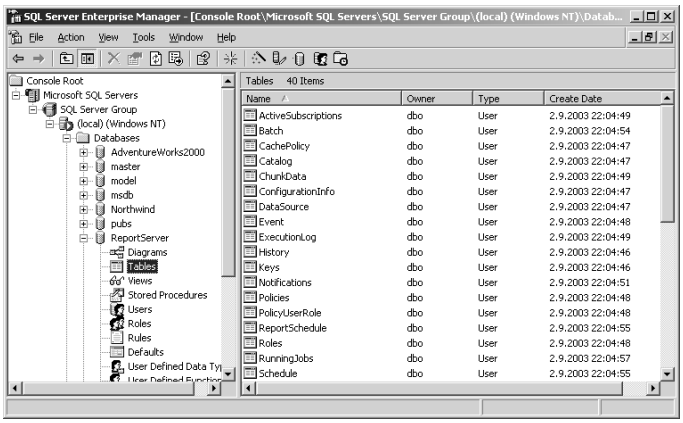


Obr. 1.3 Report Manager – prístup cez webový prehliadač

Report Manager je prístupný aj cez menu operačného systému *Start – Programs – Microsoft SQL Server – Report Manager*.

Databáza Report Servera

Údaje potrebné pre konfiguráciu a chod report servera sú uložené v databáze ReportServer pod správou databázového servera SQL Server 2000. Štruktúru a tabuľky tejto databázy si môžeme prezrieť napríklad pomocou nástroja „SQL Server Enterprise Manager“. Databáza report servera sa vytvorí a naplní počas inštalácie reportovacích služieb



Obr. 1.4 Databáza Report Servera

Report Designer

Report Designer sa spúšťa v prostredí Visual Studio .NET 2003. Tento nástroj slúži pre vytváranie editovanie a prehliadanie reportov. Po ukončení návrhu môžeme report poslať pod správu report servera.

Obr. 1.5 Report Designer v prostredí Visual Studio .NET 2003

Documentácia a príklady

Dokumentácia je nainštalovaná v adresári:

```
C:\Program Files\Microsoft SQL Server\MSSQL\ReportingServices\Help\<languagefolder>.
```

Príklady sú nainštalované v adresári

```
C:\Program Files\Microsoft SQL Server\MSSQL\Reporting Services\Samples
```

Ak sme ich nenainštalovali pri inštalácii reportovacích služieb, môžeme ich nainštalovať kedykoľvek dodatočne spustením súboru

```
sql2000reportingservicessamples.msi
```

Príprava na inštaláciu

Predtým, než začneme inštalovať reportovacie služby, potrebujeme mať nainštalované

Web Server	Windows 2000 prípadne Windows2003 Server Internet Information Server .NET Framework SMTP Server pre doručovanie e-mailov
Databázový Server	SQL Server 2000 so servicepackom 3A SQL Server Agent
Ostatné aplikácie	Visual Studio .NET 2003 .NET Framework (je zahrnutý v inštalácii vývojového prostredia)

Obr. 1.6 Inštalácia Reportovacie služby vyžadujú Service Pack 3a, prípadne vyšší

Požiadavky pre inštaláciu klientských komponentov:

- Windows 2000 všetky verzie
- Windows XP Professional so SP1 prípadne neskoršie verzie

Požiadavky pre inštaláciu serverových komponentov:

- Windows 2000 Server + SP4,
- Windows 2000 Advanced Server + SP3
- Windows 2000 Datacenter Server + SP4
- Windows Server 2003, Standard Edition, Enterprise Edition a Datacenter Edition

Podmienkou je samozrejme nainštalovaný databázový server SQL Server 2000 + SP3 (u verzie Standard musí byť na lokálnom PC, Enterprise verzia umožňuje prístup na vzdialený server) a webový server IIS verzia 5.0 prípadne novší.

Podmienkou pre nainštalovanie Report Designera je vývojové prostredie Visual Studio .NET 2003, ktoré musí byť na počítači nainštalované ešte pred zahájením inštalácie Report Designera

Podporované typy prehliadačov HTML stránok

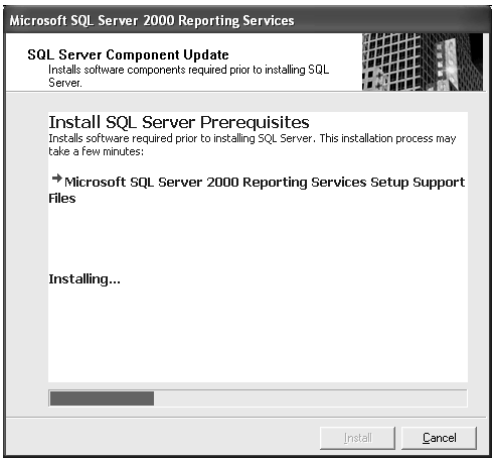
- Microsoft Internet Explorer 6.0 + SP1
- Microsoft Internet Explorer 5.5 + SP2
- Microsoft Internet Explorer 5.01 + SP2
- Netscape 7.01
- Netscape 4.78

Pre inštaláciu reportovacích služieb potrebujeme nasledovné privilégia

- Create logins
- Create roles
- Create databases
- Assign permissions to users

Proces inštalácie

Pred vlastnou inštaláciou sa ako prvý krok aktualizujú niektoré komponenty SQL Servera 2000.

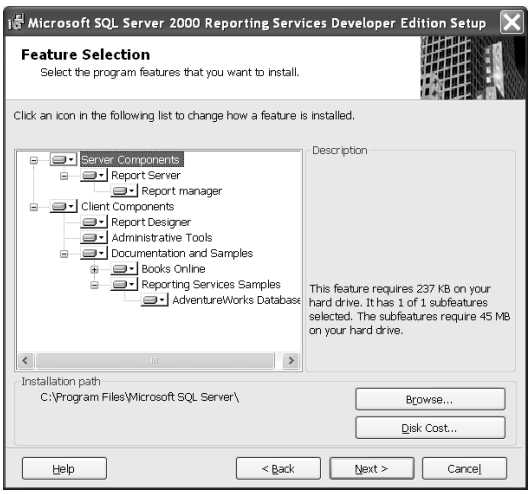


Obr. 1.7 Reportovacie služby – inštalácia komponentov pre SQL Server 2000

Po úspešnej aktualizácii komponentov začína samotný inštalačný proces reportovacích služieb. Inštalačný proces je štandardný a väčšinu parametrov môžeme ponechať implicitne nastavených.

Obr. 1.8 Reportovacie služby – sprievodca inštaláciou

Rozhodujúcim krokom je výber komponentov, ktorých inštaláciu požadujeme. Rozvaha o účele a použití jednotlivých komponentov je v predchádzajúcej stati. Kompletná inštalácia vrátane nápovedy zaberie zhruba 320 MB



Obr. 1.9 Inštalácia – výber komponentov pre inštaláciu

Reportovacie služby ako služba operačného systému Windows vyžaduje pre svoju inštaláciu a administráciu určité prístupové privilégiá a systémové účty.

Obr. 1.10 Inštalácia – výber administrátorského účtu

Počas inštalácie budú vytvorené dva virtuálne adresáre. Jeden pre reportovací server s URL adresou

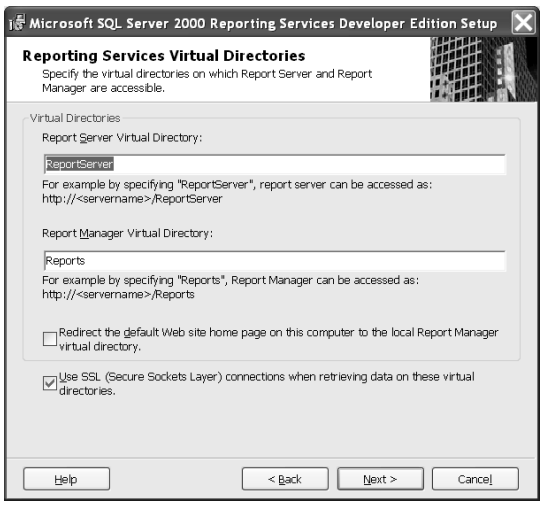
http://<servername>/ReportServer.

a jeden pre prístup cez webový server s URL adresou

http://WebServerName/Reports.

Po nainštalovaní reportovacích služieb si pomocou nástroja IIS Manager (je v zložke administrátorských nástrojov operačného systému dostupnej cez menu START - Control Panel), môžeme overiť jednak vytvorenie týchto virtuálnych adresárov a taktiež môžeme zistiť, ktoré fyzické adresáre zodpovedajú spomínaným dvom virtuálnym adresárom. Implicitne sú to adresáre:

reports	C:\Program Files\Microsoft SQL Server\MSSQL\Reporting Services\app
reportserver	C:\Program Files\Microsoft SQL Server\MSSQL\Reporting Services\ReportServer



Obr. 1.11 Inštalácia – nastavenie virtuálnych adresárov

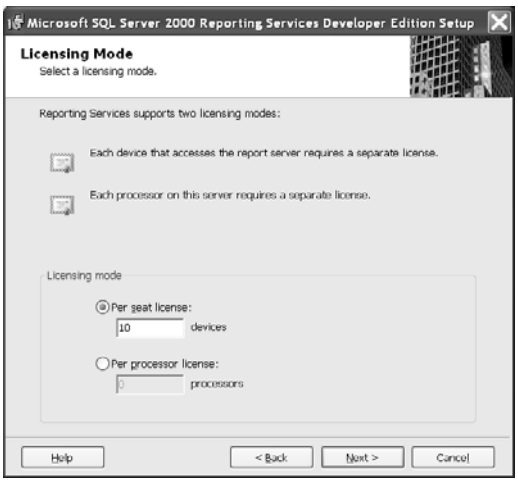
V nastavovaní prístupových parametroch pokračujeme nastavením typu prístupu

Obr. 1.12 Inštalácia – nastavenie typu prístupu

Reportovacie služby umožňujú zasielanie reportov elektronickou poštou. Pre tento účel je potrebné nastaviť adresu SMTP servera a e-mailovú adresu odosielateľa.

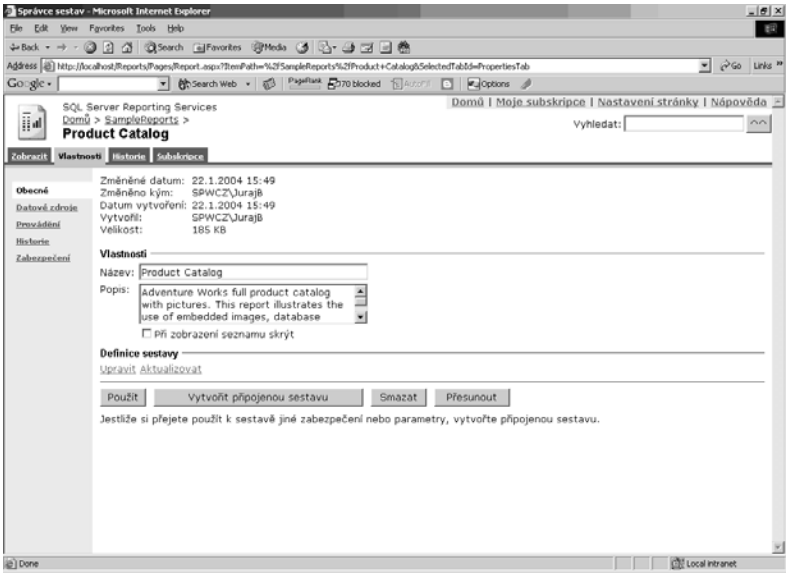
Obr. 1.13 Inštalácia – nastavenie SMTP servera a emailovej adresy pre doručovanie reportov

Ako posledný sa nastavuje licenčný mód. Na výber sú dva licenčné módy
– Per Seat
– Processor License



Obr. 1.14 Inštalácia – nastavenie licenčného módu

Reportovacie služby, konkrétne ich časť Report Manager je lokalizovaná do češtiny a možno bude aj slovenská verzia

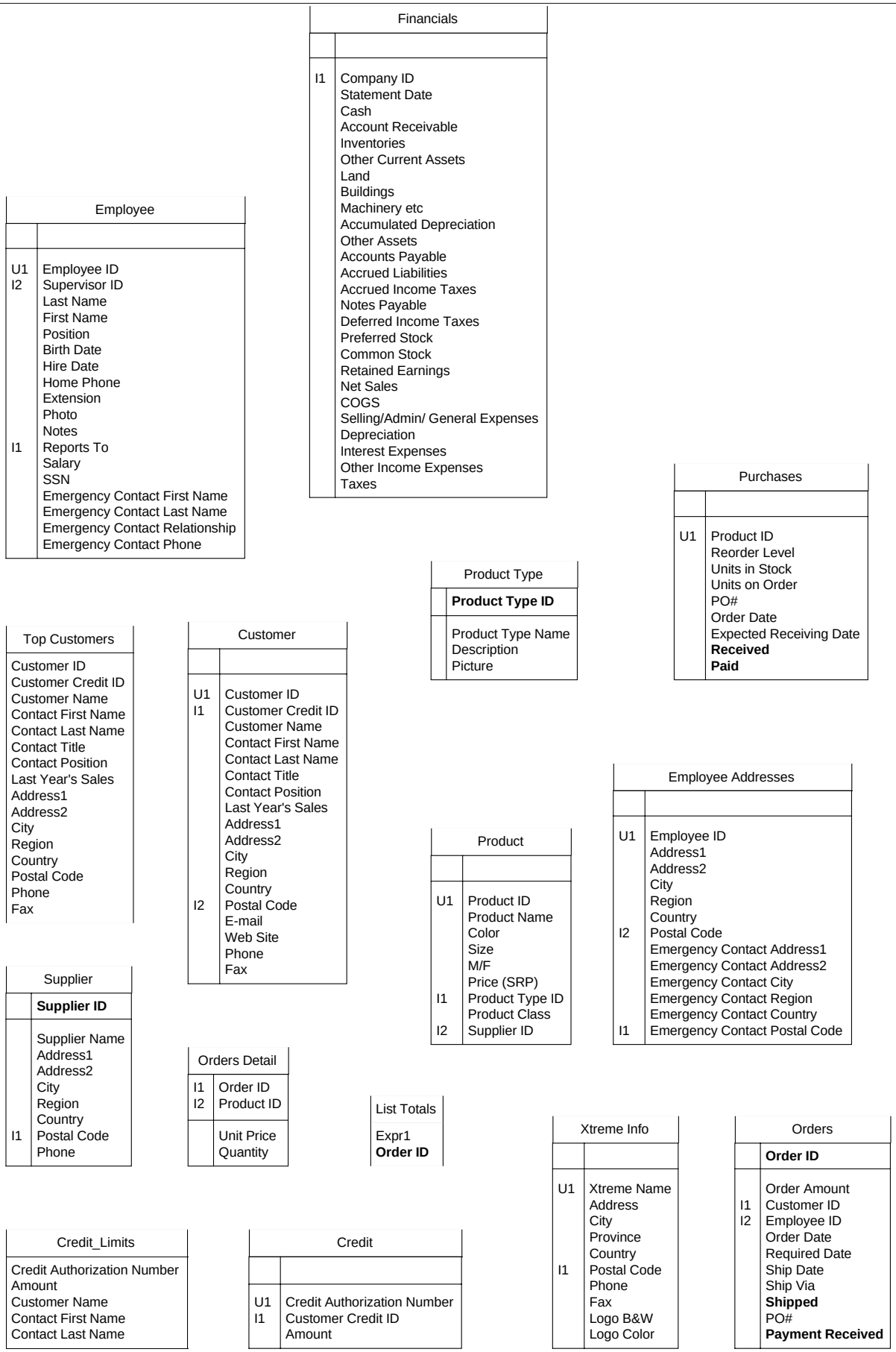


Štruktúra cvičnej databázy AdventureWorks2000

Spolu s reportovacími službami sa dodávajú aj cvičné príklady, ktoré pracujú nad údajmi z cvičnej databázy AdventureWorks2000. Ak pri výbere komponentov pre inštaláciu zaškrtneme voľbu AdventureWorks2000, nainštaluje sa aj táto databáza fiktívnej firmy s názvom Adventure Works Cycles, ktorá obchoduje s bicyklami, náhradnými súčiastkami a príslušenstvom k bicyklom a cyklistickým športovým oblečením. Databáza obsahuje údaje o produkcii, predaji, ľudských zdrojoch a nákupoch do firmy.

Štruktúra databázy AdventureWorks2000

Miesto podrobného popisu databázy, jednotlivých tabuliek a väzieb medzi nimi uvádzame len diagram popisujúci návrhovú štruktúru jednotlivých tabuliek. Záujemcovia môžu databázu pochopiteľne preskúmať podrobnejšie pomocou aplikácií Query Analyzer a Enterprise Manager



Obr. 1.15 Schema niektorych dôležitých tabuliek a pohľadov databázy AdventureWorks2000

Kapitola 2:

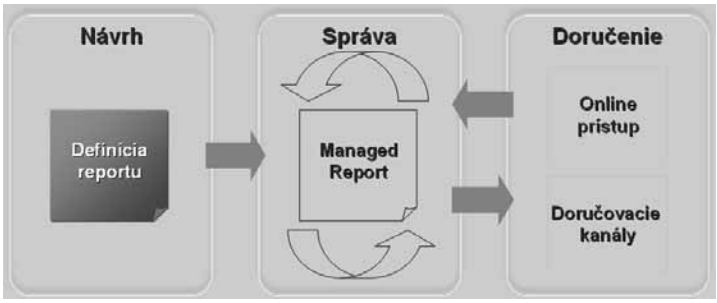
Architektúra a filozofia produktu

Filozofiu nasadenia reportovacích služieb najlepšie pochopíme ak sa zamyslíme nad životným cyklom reportu. Začneme určitou analógiou z ktorej budeme vychádzať – životným cyklom klasickej aplikácie. životný cyklus aplikácie začína vývojom, no vývoj je len malá, aj keď významná časť celého cyklu. Do etapy vývoja by sme mohli čiastočne zaradiť aj testovanie softvéru. Po otestovaní je potrebné aplikáciu pomocou vhodného marketingového kanálu a vhodného technického alebo komunikačného prostriedku distribuovať ku koncovým používateľom. Aplikácie sa šíria rôznymi druhmi distribučných kanálov, pričom ako médiá prevládajú v dnešnej dobe CD a DVD. Menšie aplikácie je možné distribuovať aj prostredníctvom internetu. Po úspešnej (zatiaľ) distribúcii produktu k zákazníkom by si mohol jeho dodávateľ spokojne mädliť ruky a oddávať sa sladkej nečinnosti, prípadne naplno sa venovať vývoju nejakej inej – novej aplikácie. Ovšem nielen vo fyzike, ale aj v informatike platí entrópia – zjednodušene povedané veci ponechané samy na seba spejú od desiatich k piatim. A tak softverový produkt, ktorý je ponechaný bez podpory, časom chátra, chradne, až nakoniec zanikne. Azda najviac životaschopné sú v prípade ponechania samých na seba paradoxne ... počítačové vírusy.

Podobne je to aj so životným cyklom reportu. Návrh reportu by sme mohli porovnať s vývojom aplikácie. Report v etape návrhu je len akýmsi predpisom, ktorý definuje k akým údajom a akým spôsobom sa bude pristupovať. Až po otestovaní a nasadení bude report generovať požadované údaje jednak na základe informácií, ktoré si nesie v svojom kóde a jednak na základe požiadaviek používateľa. Ani v tejto etape nemôžeme ponechať proces bez „údržby“. Azda najvýstižnejšou analógiou je administrácia databázového servera. V životnom cykle reportu už zostáva len jedna maličkosť. Doručiť report vhodným spôsobom v požadovanom čase a požadovanom objeme klientovi. Ak by sme zhrnuli predchádzajúci odstavec, dospeli by sme k trom fázam životného cyklu reportu

- Návrh
- Správa
- Doručenie

Ako vyplýva z obrázku, medzi poslednými dvomi etapami, teda správou reportu a procesom jeho doručovania je obojsmerná interakcia.



Obr. 2.1 Životný cyklus reportovania

Rozoberme si jednotlivé etapy trochu podrobnejšie.

Návrh reportu

V tejto etape vývojári navrhujú reporty, ktoré budú následne publikované prostredníctvom Report Servera. Pre vývoj reportu je možné využiť nástroje Microsoftu (Report Designer integrovaný do vývojového prostredia Visual Studio .NET 2003) prípadne nástroje tretích strán, ktoré budú vývoj reportov podporovať. Reporty sa definujú v jazyku RDL (Report Definition Language). Kód v jazyku **RDL** sa zapisuje vo forme XML dokumentu. XML samozrejme nie je cieľom, ale len formou. Podobne je to s kódom v ľubovoľnom programovacom jazyku, ktorý vytvárame v textovom editore. Textový dokument je pre zdrojový kód len forma zápisu. Podobne XML je pre RDL kód je len forma zápisu. Pre ilustráciu ukážeme kód prázdneho reportu

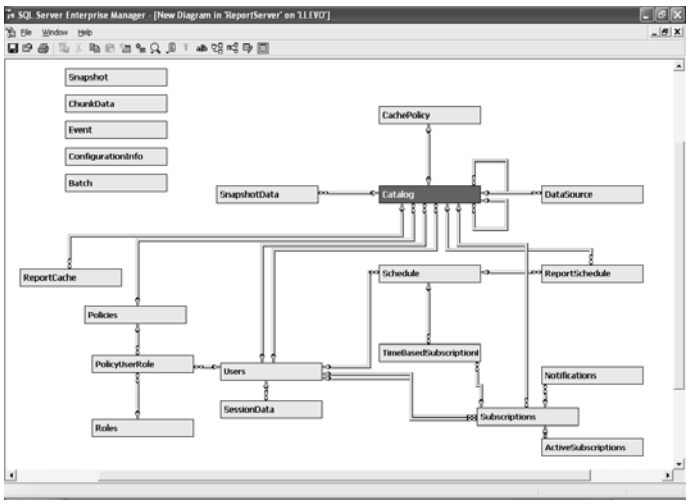
```
<?xml version="1.0" encoding="utf-8"?>
<Report xmlns="http://schemas.microsoft.com/sqlserver/reporting/2003/04/reportdefinition">
       <Width>5in</Width>
       <Body>
              <Height>2in</Height>
       </Body>
```

```
        <LeftMargin>1in</LeftMargin>
        <RightMargin>1in</RightMargin>
        <TopMargin>1in</TopMargin>
        <BottomMargin>1in</BottomMargin>
</Report>
```

Údaje prezentované prostredníctvom reportov môžu byť zobrazované v nespočetných variáciách tabuliek matíc grafov, či hypertextu. Variabilnosť podporujú dynamické a hierarchické parametre, možnosť utriedenia, filtrovania, zoskupovania údajov a výpočet čiastkových výsledkov pomocou agregačných funkcií. Hierarchicky usporiadané údaje môžeme zobrazovať na rôznych úrovniach hierarchickej štruktúry og globálnej až po veľmi detailný výsek množiny údajov. Pohyb v hierarchickej štruktúre (Drill-down, Drill-through) podstatným spôsobom rozširuje interaktívnosť a flexibilitu reportov. Flexibilný je aj základný kameň, na ktorom je každý report postavený – SQL dopyt. Zmenou dopytu môžeme taktiež dosiahnuť vysokú variabilitu. Dopytovanie je možné nielen do databáz (či už relačných alebo analytických) pod správou SQL Servera a analytického servera, ale aj do iných databáz, napríklad Oracle, prípadne do databáz podporujúcich ODBC alebo OLEDB rozhrania.

Správa reportu

V tejto etape spravujeme a zlaďujeme návrhy reportov, adresáre, resource a podobne. Riadené (managed) reporty môžu byť reportovacou službou vygenerované buď aktuálne „na požiadanie“ prípadne na základe nejakých plánov a časových rozvrhov. Výkonnosť reportovacích služieb samozrejme podstatnou mierou zvyšuje správne kešovanie, nakoľko je predpoklad, že niektoré typy reportov budú vhodné pre viacerých klientov. Pre správu reportov je možné využiť API webovej služby, prípadne používateľské rozhrania pre Web a Win32. Metadáta (údaje o údajoch) sú uložené vo vlastnej databáze reportovacích služieb, ktorá je taktiež pod správou SQL Servera.



Obr. 2.2 Diagram vnútornej databázy reportovacích služieb

Činnosti, ktoré sa vykonávajú podľa časového plánu riadi SQL Server Agent. O činnosti reportovacích služieb sa samozrejme zaznamenávajú údaje do protokolov.

Aj pojem „životný cyklus reportu“ je možné chápať dvojakým spôsobom. Jednak z hľadiska rozdelenia na jednotlivé etapy a taktiež jeho premietnutie na časovej osi. Princíp je triviálny. Ak potrebujeme report o 9:00 a pre jeho vytvorenie (napríklad na základe údajov v terabajtovom dátovom sklade) potrebujeme dve hodiny, musíme začať o 7.00, alebo ešte lepšie o niečo skôr, aby bola určitá časová rezerva. Pomocou reportov zobrazujeme údaje buď z relačných databáz, alebo z dátových skladov. V druhom prípade je potrebné rešpektovať aj životný cyklus dátového skladu.

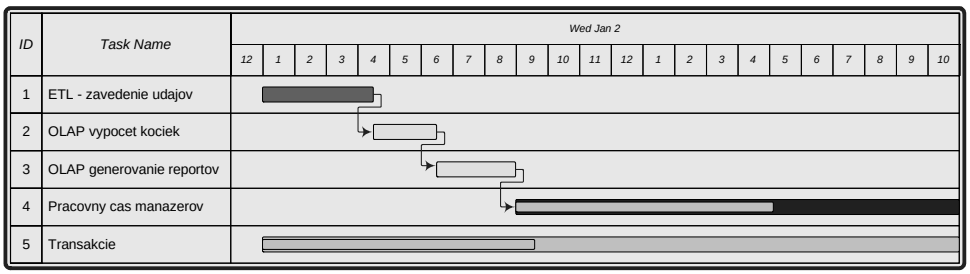
Výsledkom analýzy sú vždy súhrny a reporty, ktoré slúžia manažérom ako podklady pre ich rozhodnutia, či už v oblasti riadenia firmy, riadenia ekonomických a technologických procesov a podobne. Pripomeňme si na časovej osi životný cyklus „reportu“ ktorý je výsledkom OLAP analýzy údajov v dátovom sklade.
Upozorňujeme, že údaje v prvých troch riadkoch časového grafu sa týkajú predchádzajúceho dňa to znamená že v našom prípade sa po polnoci vždy analyzujú údaje z predchádzajúceho dňa. To sa týka predovšetkým dátového skladu. V prípade reportov z údajov v transakčných databázach môže celý proces prebiehať takmer v reálnom čase.

Pri slovnom popise celého procesu znázorneného v časovej tabuľke bude lepšie postupovať zdola nahor.

Transakcie
V poslednom riadku (5) vidíme, že transakcie dnešného dňa bežia nepretržite 24 hodín. Môže sa jednať o výrobný proces, elektronický obchod, bankové transakcie, proste OLTP proces ktorý je zdrojom údajov pre analýzy prípadne reporty. Tieto analýzy dnešných údajov sa budú vykonávať až po polnoci, aby zajtra na začiatku pracovnej doby boli k dispozícii. Reporty môžeme niekedy generovať priamo „na vyžiadanie“.

Pracovný čas manažérov
Začiatok pracovného času manažérov (v našej tabuľke 09:00) je rozhodujúcim okamihom, kedy by mali byť výsledky analýzy a príslušné reporty týmto manažérom k dispozícii. Na grafe je zohladnený aj fakt, že väčšina manažérov sú hlavne v krízovejších obdobiach zároveň aj workoholikmi, alebo to aspoň o sebe tvrdia.

Proces prípravy údajov a analýza
Aby mohli manažéri získať v stanovenom čase kvalitné podklady pre podporu rozhodovania, musí prebehnúť príprava údajov pre analýzu, v prípade warehousingu ich zavedenie do dátového skladu a ich následná analýza. V našom príklade proces ETL *(riadok 1 časovej tabuľky)* začne ihneď po polnoci o 00:00. Po zhruba troch hodinách je táto etapa ukončená a môže začať napočítavanie kociek a ostatné agregačné a analytické výpočty *(riadok 2 časovej tabuľky)*. Na tento proces nadviaže generovanie zostáv, reportov a podobne *(riadok 3 časovej tabuľky)*.



Obr. 2.3 Cyklus OLAP reportu na časovej osi

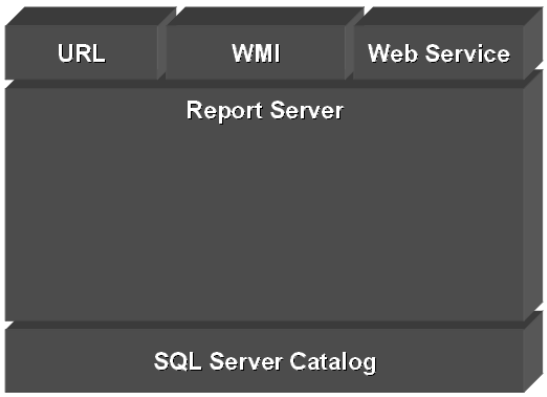
Doručenie reportu
V tejto etape je dôležitý jednak spôsob doručenia (prípadne doručovania) a jednak forma. Reporty sa môžu generovať a doručovať priamo „na požiadanie“ alebo na základe časových plánov a posielať napríklad mailom. Pod formou budeme rozumieť tabuľky, grafy, obchodnú grafiku a podobne. Taktiež si môžeme zvoliť druh dokumentu, alebo technicky povedané výstupný formát do ktorého sa report vyrendruje. K dispozícii sú:

- Webové formáty (HTML 4, HTML 3.2, HTML w/OWC)
- Printovateľné formáty (TIFF, RTF, PDF)
- Dátové formáty (Excel, XML, CSV)

Odberateľ môže dostávať reporty vo forme mailov, prípadne pomocou zdieľania súborov (file share)
Odberatelia reportov si samozrejme môžu individuálne personalizovať reporty, ktoré majú dostávať

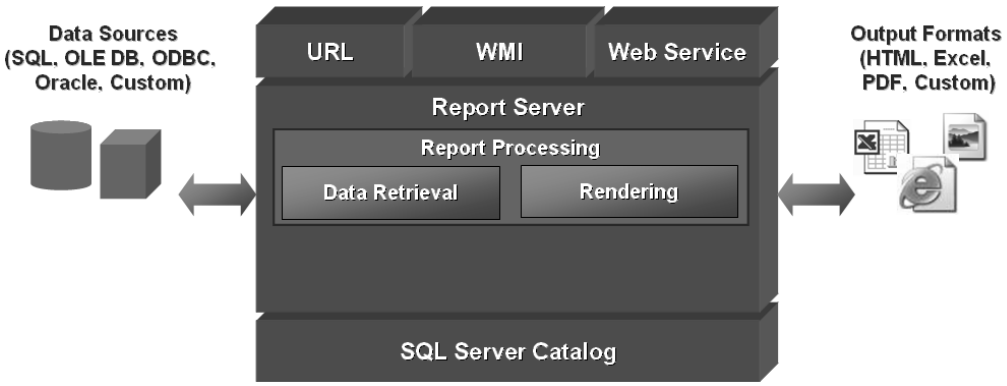
Architektúra reportovacích služieb

Architektúra reportovacích služieb je niekoľkovrstvová. Základnú vrstvu tvorí SQL Server Catalog, čo je databáza pod správou SQL Servera. Údaje z tejto databázy využíva Report Server pre ukladanie metadát, snapshotov, definícií reportov, adresárov, údajov pre zabezpečenie a podobne. Dôležitou informáciou je to, že reportovacie služby samotné sú bezstavové. Nad vrstvou Report server je vrstva primárnych aplikačných rozhraní URL, WMI a rozhranie pre webové služby. Rozhranie WMI slúži pre správu reportovacích služieb.



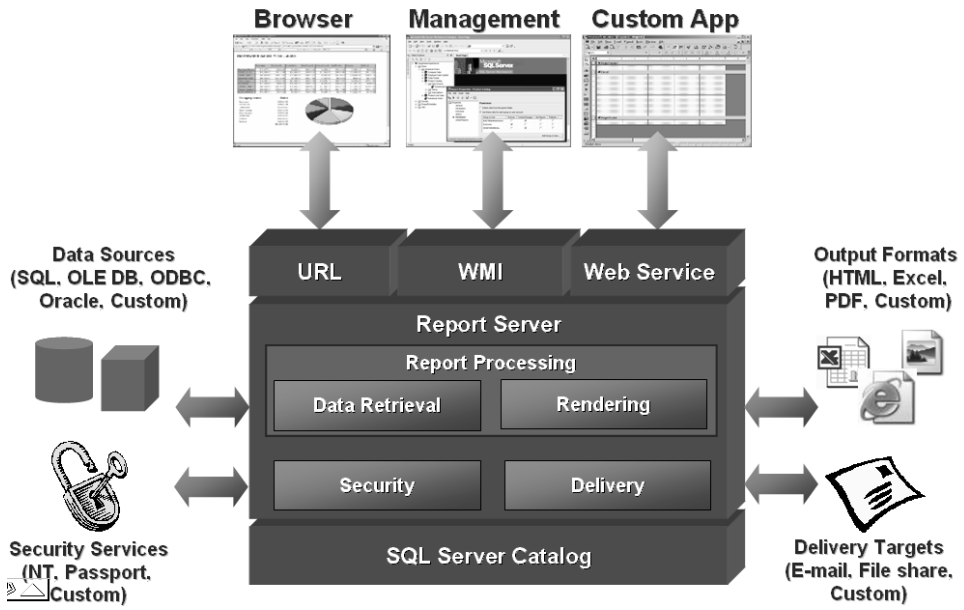
Obr. 2.4 Základná vrstva architektúry

Vo vstve Report Server prebiehajú rôzne procesy, ktorých cieľom je na základe návrhu reportu a údajov vyrendrovať report v požadovanom výstupnom formáte



Obr. 2.5 Základná vrstva architektúry

Na primárne aplikačné rozhrania sú naviazané aplikácie. Na URL rozhranie pristupujeme pomocou prehliadača webového obsahu, napríklad Internet Explorera. Cez WMI rozhranie pristupujeme z administátorskej konzoly, napríklad MS Operation Manage, HP Open View a podobne.
Bežné aplikácie využívajú rozhranie webovej služby



Obr. 2.6 Základná vrstva architektúry

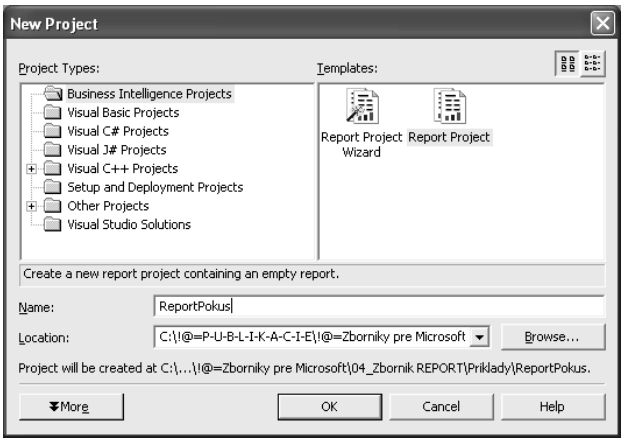
Kapitola 3:

Reportovacie služby z pohľadu vývojára

V tejto kapitole si môžeme dovoliť strohý technický štýl a tak hneď môžeme začať faktami. Report sa navrhuje v jazyku RDL (Report Definition Language). Tento kód sa však nezapisuje do textového súboru ako je to bežné u zdrojových kódov vyšších programovacích jazykov, ale do XML dokumentu. XML dokument je totiž pomerne optimálny kompromis zrozumiteľnosti obsahu dokumentu jednak pre ľudí a jednak pre počítače. Zatiaľ čo zdrojový alebo skriptový kód zapísaný v textovom súbore môže mať pomerne voľnú formu, čo je výhodné pre ľudí (každý si zvolí štýl a úpravu, ktorá mu vyhovuje) ale menej výhodné pre počítače, ktoré musia zdrojový kód v textovom dokumente pomerne náročne analyzovať. XML dokument má naopak pevnú formálnu štruktúru a na rozdiel od napríklad binárneho súboru jeho zrozumiteľnosť pre človeka príliš neklesá.

Pre návrh reportu máme niekoľko možností. Môžeme použiť Report Designer ktorý sa po nainštalovaní reportovacích služieb integruje do vývojového prostredia Visual Studio .NET 2003, prípadne nástroje pre návrh od tretích strán, ktoré sa vďaka otvorenej štruktúre reportovacích služieb určite čoskoro objavia. Pardon, skoro by sme boli zabudli na tretiu možnosť, a síce, že nebudeme využívať žiadny vizuálny nástroj, ale kód v RDL jazyku napíšeme priamo vo svojom obľúbenom editore XML dokumentov.

Najbežnejším spôsobom návrhu bude zrejme Report Designer integrovaný do vývojového prostredia Visual Studio .NET 2003. Na rozdiel od stavu pred inštaláciou reportovacích služieb sa v dialógu pre vytvorenie nového projektu objaví zložka Business Intelligence Projects.



Obr. 3.1 Dva spôsoby pre zahájenie návrhu reportu Report Wizard a Blank Report

Pre zahájenie návrhu nového projektu máme dve možnosti. Môžeme využiť sprievodcu vytvorením reportu (**Report Wizard**) alebo pri vytvorení nového projektu sa vytvorí prázdny report (**Blank Report**)

Anatómia reportu, základy jazyka RDL

Ako sme už zdôraznili kód návrhu reportu v jazyku RDL je uložený v XML dokumente. Preto skôr než sa dostaneme k základom jazyku RDL (tento jazyk je mimochodom veľmi intuitívny a ak budeme používať vizuálne vývojové prostredie, budú nám skutočne stačiť len úplné základy) sa budeme venovať základným princípom jazyka XML

XML – formát budúcnosti

Pri hľadaní vhodného formátu pre výmenu údajov hlavne medzi rôznymi aplikáciami a na rôznych platformách pre e-business (teda populárne B2B aplikácie) sa zistilo, že týmto požiadavkám najviac vyhovuje jazyk XML (Extensible Markup Language). Podporu XML majú implementované všetky moderné databázové servery. Pod pojmom XML dokument rozumieme dokument, napríklad súbor s príponou XML, ktorého obsah spĺňa pravidlá syntaxe značkovacieho jazyka XML. Dokument sa skladá z elementov. Každý element obsahuje počiatočnú a koncovú značku, inými slovami počiatočný a koncový tag. Obidva tagy obsahujú názov elementu, koncový tag obsahuje navyše pred názvom elementu lomítko.

```
<meno>Jan Novak</meno>
```

Každý element môže v sebe obsahovať ďalšie takzvané vnorené elementy.

```
<zakaznik>
    <meno>Jan Novak</meno>
    <adresa>Hladikova 4</adresa>
    <mesto>Bratislava</mesto>
</zakaznik>
```

Vidíme, že XML dokument má hierarchickú stromovú štruktúru. Úroveň vnorenia jednotlivých elementov nie je obmedzená, vyžaduje sa však, aby každý XML dokument mal jeden koreňový (ROOT) element

<root_element>
...obsah dokumentu
</root_element>

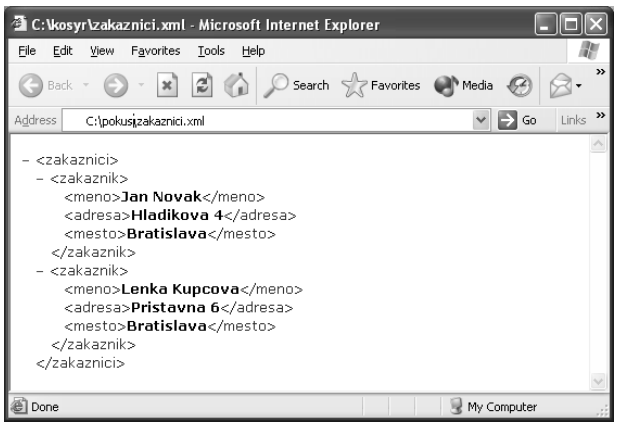
takže ak by sme chceli uložiť do XML dokumentu údaje o viacerých zákazníkoch, zapúzdrili by sme jednotlivé záznamy do koreňového elementu `<zakaznici>..</zakaznik>`

```
<zakaznici>
  <zakaznik>
      <meno>Jan Novak</meno>
      <adresa>Hladikova 4</adresa>
      <mesto>Bratislava</mesto>
  </zakaznik>
  <zakaznik>
      <meno>Lenka Kupcova</meno>
      <adresa>Pristavna 6</adresa>
      <mesto>Bratislava</mesto>
  </zakaznik>
</zakaznici>
```

Rozumné je aj doplniť deklaráciu použitej verzie jazyka XML (nie je povinná):

```
<?xml version="1.0"?>
<zakaznici>
..
</zakaznici>
```

Pre naše účely táto krátka exkurzia do tajov XML dokumentu úplne postačuje. Azda najdostupnejším programom pre zobrazenie obsahu XML súboru môže byť prehliadač webových stánok Internet Explorer



Obr. 3.2 Internet Explorer ako prehliadač XML súborov

Jazyk RDL – štruktúra

Koreňové elementy, URI namespace a základné parametre prázdneho reportu sú uložené v XML dokumente v tvare

```
<?xml version="1.0" encoding="utf-8"?>
<Report xmlns="http://schemas.microsoft.com/sqlserver/reporting/2003/04/reportdefinition"
xmlns:rd="http://schemas.microsoft.com/SQLServer/reporting/reportdesigner">
...
  <Body>
    <Height>2in</Height>
  </Body>
  <RightMargin>1in</RightMargin>
  <TopMargin>1in</TopMargin>
  <rd:SnapToGrid>true</rd:SnapToGrid>
</Report>
```

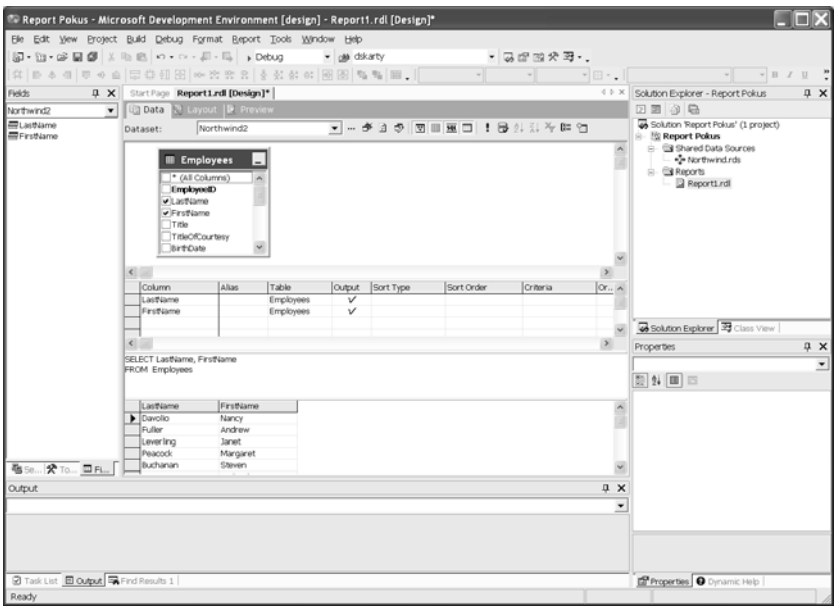
Report, respektíve súbor s kódom jeho návrhu obsahuje tri druhy informácií
- **Údaje,** prípadne informácie ako sa k požadovaným údajom pripojiť (najčastejšie SQL dopyt), štruktúry údajov
- **Návrhová schéma,** ktorá obsahuje informácie o tom v akej podobe sa údaje budú prezentovať.
- **Vlastnosti** reportu a jeho jednotlivých prvkov.

Informácie o údajoch ku ktorým sa budeme pripájať zadávame už v začiatku etapy návrhu, kedy definujeme pripojenie sa k ich zdroju. V našom ilustračnom príklade (podrobný postup bude vysvetlený pri cvičných príkladoch) sa pripojíme k cvičnej databáze Northwind, z ktorej nás bude zaujímať tabuľka Employees, konkrétne stĺpce FirstName a LastName. Predchádzajúcu vetu dokážeme stručne vyjadriť pomocou SQL príkazu

```
SELECT  LastName, FirstName FROM  Employees
```

ktorý bude aj základom pre výber údajov.

Vo vývojovom prostredí situáciu vysvetľuje obrázok.



Obr. 3.3 Vizuálny návrh datasetu

V kompletnom kóde reportu (prázdneho, obsahujúceho len definíciu údajov a pripojenie na ich zdroj) si všimnime časti označené hrubým fontom. Týkajú sa datasetu, (sekcia **<DataSets>**) ktorý vznikne na základe nami definovaného SQL príkazu a sekcie pre definovanie pripojenia sa k zdroju údajov **<DataSources>**

```
<?xml version="1.0" encoding="utf-8"?>
<Report xmlns="http://schemas.microsoft.com/sqlserver/reporting/2003/04/reportdefinition"
xmlns:rd="http://schemas.microsoft.com/SQLServer/reporting/reportdesigner">
  <DataSets>
    <DataSet Name="Northwind">
      <Fields>
        <Field Name="LastName">
          <rd:DataSetName>Northwind</rd:DataSetName>
          <Alias>LastName</Alias>
          <rd:TypeName>System.String</rd:TypeName>
        </Field>
        <Field Name="FirstName">
          <rd:DataSetName>Northwind</rd:DataSetName>
          <Alias>FirstName</Alias>
          <rd:TypeName>System.String</rd:TypeName>
        </Field>
      </Fields>
      <Query>
        <DataSourceName>Northwind</DataSourceName>
        <CommandType>Text</CommandType>
        <CommandText>SELECT  LastName, FirstName FROM  Employees</CommandText>
        <Timeout>30</Timeout>
      </Query>
    </DataSet>
  </DataSets>
  <Width>5in</Width>
  <rd:DrawGrid>true</rd:DrawGrid>
  <BottomMargin>1in</BottomMargin>
  <LeftMargin>1in</LeftMargin>
  <DataSources>
    <DataSource Name="Northwind">
      <ConnectionProperties>
        <Extension>SQL</Extension>
        <ConnectString>initial catalog=Northwind;integrated security=SSPI;persist
security info=False</ConnectString>
        <IntegratedSecurity>true</IntegratedSecurity>
      </ConnectionProperties>
    </DataSource>
  </DataSources>
  <Body>
    <Height>2in</Height>
  </Body>
  <RightMargin>1in</RightMargin>
  <TopMargin>1in</TopMargin>
  <rd:SnapToGrid>true</rd:SnapToGrid>
</Report>
```

Štruktúru kódu, teda vlastne hrubú osnovu syntaxe jazyka RDL môžeme zapísať aj pomocou diagramu. Pri zápise syntaxe sa používajú rôzne formy diagramov. Nakoľko jazyk RDL najviac používajú ludia, ktorí pracujú s databázami, aj pre syntaktické diagramy sú použité entitno – relačné schémy.

Entita *(angl. entity)* je objekt reálneho sveta, ktorý je schopný nezávislej existencie a je jednoznačne odlišný od ostatných objektov.

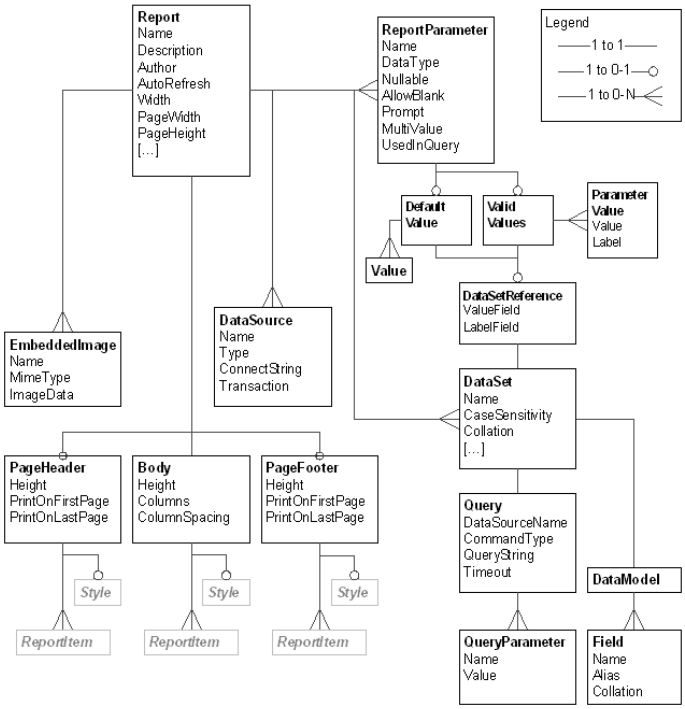
Entitno – relačný model (E-R model) je množina pojmov, pomocou ktorých popisujeme príslušnú aplikáciu za účelom následnej špecifikácie návrhovej štruktúry, v našom prípade štruktúry reportu. Proces návrhu systému spočíva jednak v identifikácii typov entít ako množín objektov rovnakého typu, v identifikácii typov vzťahov, do ktorých budú entity vstupovať a v priradení atribútov, ktoré bližšie popisujú vlastnosti jednotlivých entít a vzťahov.

Vzťah *(angl. relationship)* je väzba medzi dvoma alebo viacerými entitami.

Atribút *(angl. attribute)* je funkcia, ktorá priraďuje jednotlivým entitám alebo vzťahom hodnotu. Táto hodnota určuje niektorú podstatnú vlastnosť entity alebo vzťahu.

Tých pár riadkov entitno – relačnej teórie bolo určených len na získanie prehľadu, aby sme prípadne mohli čtať diagramy, ktoré budú nasledovať. Vymedzenie pojmov entita, vzťah a atribút je pomerne voľné, pričom záleží hlavne od uhla pohľadu analytika. Určitým vodítkom môže byť analógia s prirodzeným jazykom, kedy sa pre popis entít používajú podstatné mená a pre popis vzťahov slovesá.

A na záver pre upokojenie ľudí, ktorým nasledujúce diagramy nič nepovedia – netreba sa znepokojovať, každý má iný spôsob myslenia a určite všetko pochopíte na praktických ukážkach.



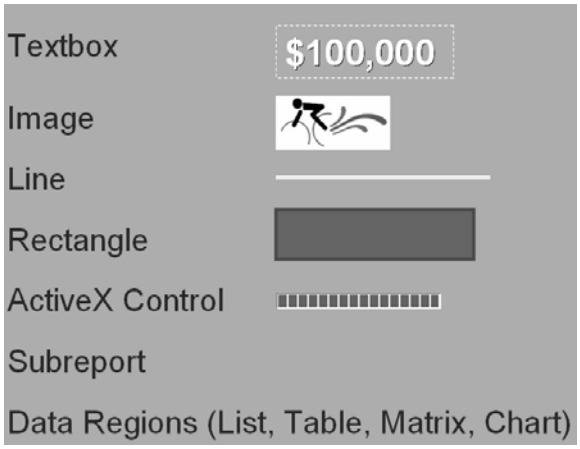
Obr. 3.4 Globálna schéma reportu v jazyku RDL

Prvky pre návrh reportov

Tu by sme sa mohli nechať zlákať vidinou bezprácneho výtvoru reportu za asistencie wizarda, no pre úspešný návrh práve takých reportov ako potrebujeme (toto je základné kritérium) budeme potrebovať niečo vedieť aj o ovládacích prvkoch z ktorých sa reporty skladajú a o ich vlastnostiach. Pre návrh reportov môžeme použiť niektoré komponenty a ovládacie prvky, ktoré poznáme napríklad z webových formulárov ASP.NET aplikácií a podobne. Sú to prvky

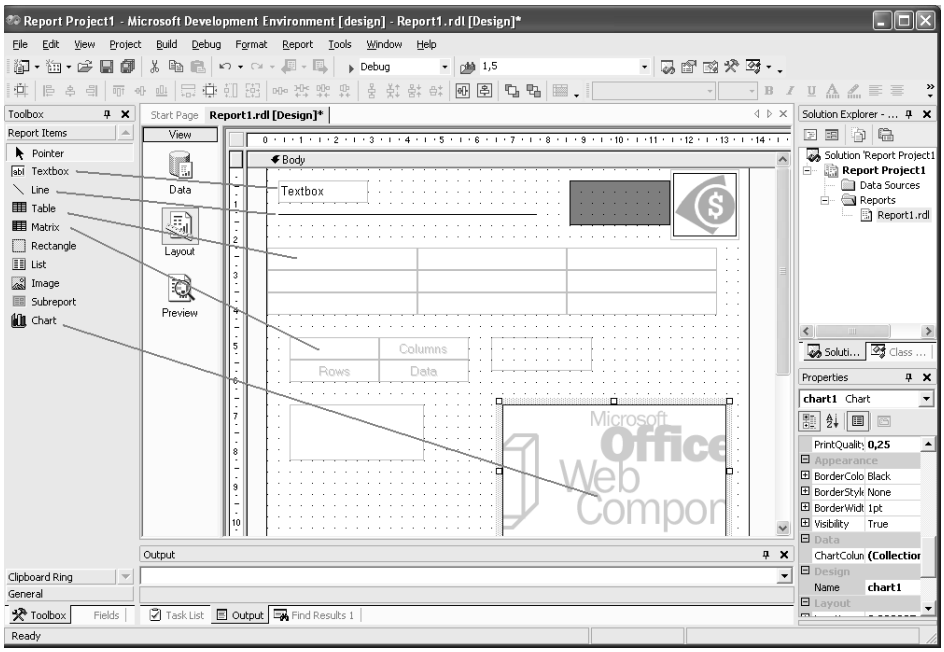
- Textbox
- Image
- Line
- Rectangle
- ActiveX Control
- Subreport
- Data Regions (List, Table, Matrix, Chart)

Prvok Textbox slúži pre výpis jednoduchých textov na určené miesto reportu. Pre grafické dotvárnenie „vizáže“ reportu slúžia prvky Image, Line a Rectangle. Jednotlivé prvky predstavíme podrobnejšie v ďalšom texte. Zatiaľ viac než popis možno napovie ilustračný obrázok a následné príklady



Obr. 3.5 Prvky reportov

V etape vlastného návrhu jednotlivé prvky, ktorých ikony sú umiestnené v toolboxe rozmiestňujeme na vhodné miesta návrhového formulára reportu a nastavujeme mu vhodné vlastnosti.



Obr. 3.6 Prvky reportov v návrhovom formulári vývojového prostredia

Podobne ako u webových a windows formulárov aj v tomto prípade môžeme pre jednotlivé prvky nastavovať niektoré vlastnosti, napríklad pre všetky prvky môžeme nastaviť parametre:

- Background Color
- Background Image
- Border Color
- Border Style
- Border Width
- Color
- Padding

Niektoré parametre môžeme nastaviť len pre určité prvky, napríklad pre textovo orientované prvky môžeme spravidla nastaviť

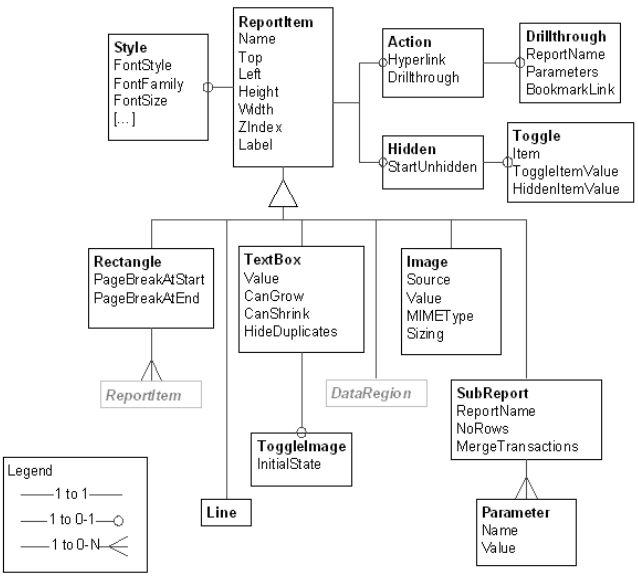
- Can Grow
- Font Family
- Font Size
- Font Style
- Font Weight
- Format
- Hide Duplicates
- Line Height
- Text Align
- Text Decoration
- Vertical Align

U väčšiny parametrov môžeme nastaviť nielen konkrétnu hodnotu, ale aj výraz v jazyku Visual Basic .NET

Údaje tom, ktoré vlastnosti môžeme nastavovať pre jednotlivé prvky obsahuje nasledovná tabuľka

	Line	Rectangle	Textbox	Image	Subreport	List	Matrix	Table	Chart
Name	+	+	+	+	+	+	+	+	+
Action			+	+					
Top	+	+	+	+	+	+	+	+	+
Left	+	+	+	+	+	+	+	+	+
Height	+	+	+	+	+	+			+
Width	+	+	+	+	+	+			+
Zindex	+	+	+	+	+	+	+	+	+
Hidden	+	+	+	+	+	+	+	+	+
Title		+	+	+	+	+	+	+	+
Label	+	+	+	+	+	+	+	+	+
LinkToChild		+							
Bookmark	+	+	+	+	+	+	+	+	+
RepeatWith	+	+	+	+					
Custom	+	+	+	+	+	+	+	+	+

Možno niekomu viac napovie syntaktický diagram



Obr. 3.7 Prvky pre návrh formulárov v schéme jazyka RDL

Iné vlastnosti zas definujú štýl, napríklad farbu, veľkosť fontu a podobne

	Line	Rectangle	Text-box	Image	Subreport	List	Matrix	Table	Chart	Body	Sub-total
BorderColor	+	+	+	+	+	+	+	+	+	+	+
BorderStyle	+	+	+	+	+	+	+	+	+	+	+
BorderWidth	+	+	+	+	+	+	+	+	+	+	+
BackgroundColor		+	+			+	+	+		+	+
BackgroundImage		+	+			+	+	+		+	+
FontStyle			+								+
FontFamily			+								+
FontSize			+								+
FontWeight			+								+
Format			+								+
TextDecoration			+								+
TextAlign			+								+
VerticalAlign			+								+
Color			+								+
PaddingLeft			+	+							+
PaddingRight			+	+							+
PaddingTop			+	+							+
PaddingBottom			+	+							+
LineHeight			+								+
CanSort			+								+
Direction			+								+
Language			+							+	+
UnicodeBiDi			+								+
Calendar			+								+
NumeralLanguage			+								+
NumeralVariant			+								+
WritingMode			+								+

Textbox

Textbox je prvok, pomocou ktorého môžeme v presne definovanom mieste reportu vypísať nejaký text. Môže to byť statický text

```
<Textbox Name="textbox1">
  <Style>
    <PaddingLeft>2pt</PaddingLeft>
    <PaddingBottom>2pt</PaddingBottom>
    <PaddingTop>2pt</PaddingTop>
    <PaddingRight>2pt</PaddingRight>
  </Style>
  <rd:DefaultName>textbox1</rd:DefaultName>
  <Height>0.25in</Height>
  <Width>1.125in</Width>
  <Top>0.952381cm</Top>
  <Value>...staticky text...</Value>
  <Left>0.6349207cm</Left>
</Textbox>
```

alebo obsah niektorej bunky z databázovej tabuľky na ktorú sa Textbox dynamicky napojí

```
<Textbox Name="LastName">
  <Style>
    ...
  </Style>
  <rd:DefaultName>LastName</rd:DefaultName>
   ...
</Textbox>
```

Pre Textbox môžeme nastavovať parametre: *Value, CanGrow, CanShrink, HideDuplicates, ToggleImage*

Line

Okrem základných parametrov ohľadne súradníc začiatočného a koncového bodu, hrúbky čiary...(všetko sa nastaví pri vizuálnom návrhu), nie je pre prvok Line potrebné nastavovať žiadne ďalšie vlastnosti

```
<Line Name="line1">
  <Style>
    <BorderStyle>
      <Default>Solid</Default>
    </BorderStyle>
  </Style>
  <Width>3in</Width>
  <Height>0in</Height>
  <Top>1.269841cm</Top>
  <Left>1.269841cm</Left>
</Line>
```

Rectangle

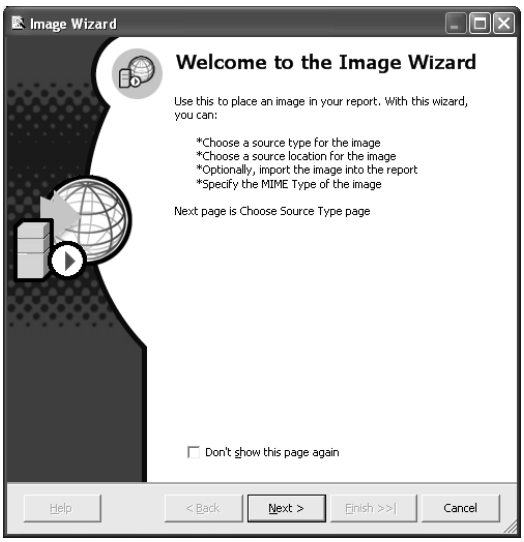
Obdĺžnik, ktorý pomáha spoluvytvárať dizajn obchodnej grafiky sa v jazyku RDL definuje nasledovne

```
<Rectangle Name="rectangle1">
  <Top>0.952381cm</Top>
  <Style>
    <BorderStyle>
      <Default>Solid</Default>
    </BorderStyle>
    <BackgroundColor>IndianRed</BackgroundColor>
  </Style>
  <Height>0.5in</Height>
  <Width>1.5in</Width>
  <Left>1.269841cm</Left>
</Rectangle>
```

Pre Rectangle môžeme nastavovať parametre: *ReportItems, PageBreakAtStart, PageBreakAtEnd*

Image

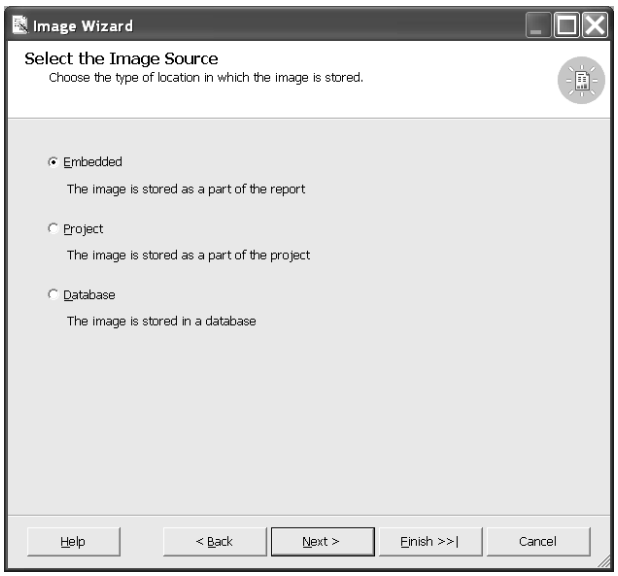
Obrázky v reporte slúžia pre oživenie a pre priblíženie problematiky zákazníkovi. Rozdiel medzi katalógom ľubovoľného druhu výrobkov s obrázkami a bez obrázkov v prospech obrázkového katalógu snáď ani netreba vysvetľovať. Pre umiestnenie obrázkov do reportu je vo vývojovom prostredí k dispozícii sprievodca.



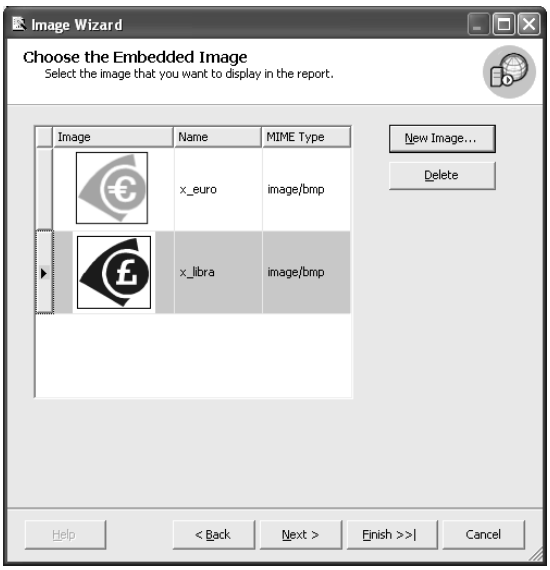
Obr. 3.8 Sprievodca pridaním obrázku do reportu

Azda najzávažnejšou otázkou, ktorú potrebujeme riešiť, je to, kam umiestniť súbor s obrázkom (resp. binárny kód, ktorý je uložený v JPG, BMP, GIF... obrazovom dokumente). K dispozícii sú tri možnosti:

Embedded – obrázok bude vložený priamo do reportu
Project – obrázok bude súčasťou projektu, teda bude uložený v databáze reportovacieho servera
Database – obrázok bude uložený v databáze z ktorej sa vytvára report



Obr. 3.9 Výber spôsobu uloženia obrázku



Obr. 3.10 Pripojovanie obrázkov k reportu

```
<Image Name="image1">
  <Top>2.222222cm</Top>
  <Style />
  <MIMEType>image/bmp</MIMEType>
  <Height>0.8020833in</Height>
  <Width>0.7916667in</Width>
  <Source>Embedded</Source>
  <Value>x_euro</Value>
  <Left>0.952381cm</Left>
  <Sizing>AutoSize</Sizing>
</Image>
```

Toto je len definícia umiestnenia obrázku. Vložený kód (skrátený) samotného obrázku je

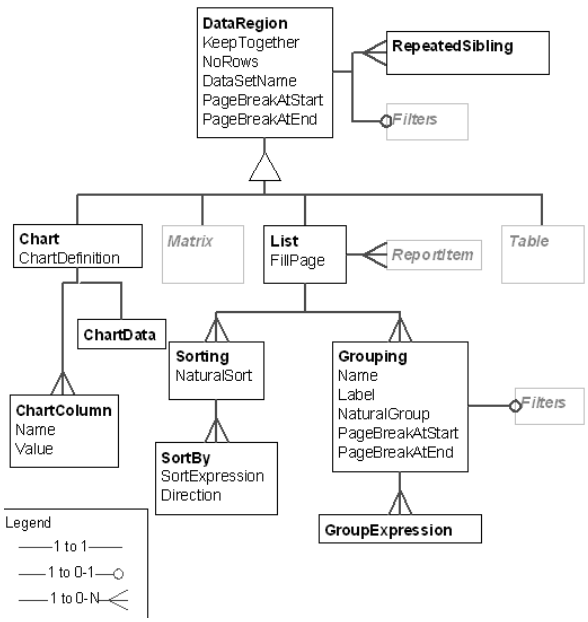
```
<EmbeddedImages>
    <EmbeddedImage Name="x_libra">
      <MIMEType>image/bmp</MIMEType>
      <ImageData>Qk3mQwAAAAAAADYAAAAoAAAAT...</ImageData>
       ...
    </EmbeddedImage>
  </EmbeddedImages>
```

Prvky pre zobrazovanie údajov

Aj keď aj čiary, obdĺžniky a obrázky tvoria dôležitú časť dizajnu obchodnej grafiky, predsa len nosnými prvkami reportov budú prvky pre zobrazovanie údajov. K dispozícii sú prvky List, Table, Matrix a Chart

Obr. 3.11 Prvky pre zobrazenie údajov

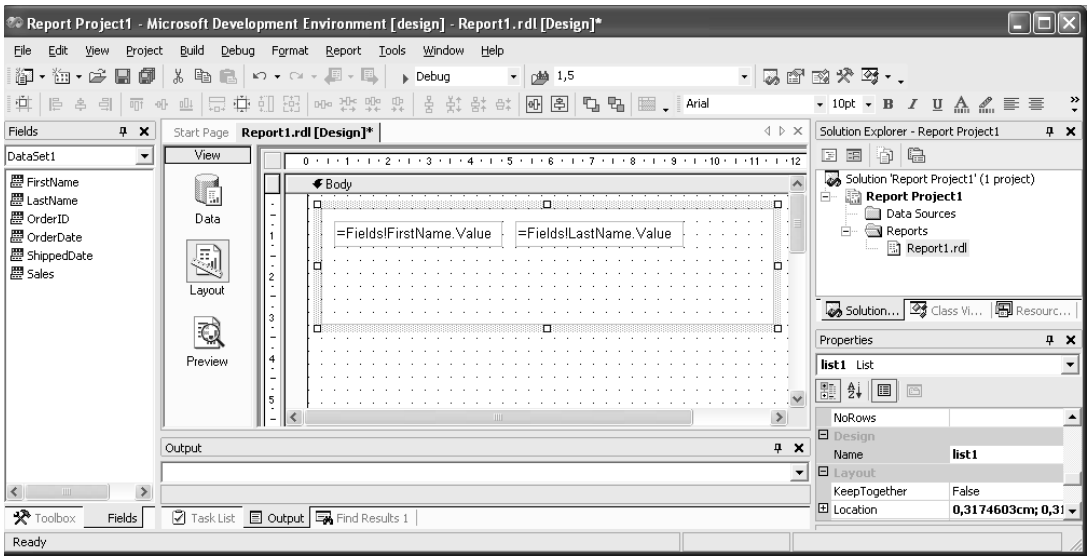
List – voľné uloženie údajov
Table – údaje sú organizované do stĺpcov
Matrix – dynamická stĺpcová organizácia údajov (kontingenčná tabuľka – pivot table)
Chart – vizualizácia údajov



Obr. 3.12 Prvky pre zobrazenie údajov v schéme jazyka RDL

List

Pomocou prvku list je možné v „pásoch“ zobrazovať opakujúce sa údaje, pričom forma zobrazenia je pomerne voľná. Vo vizuálnom návrhovom prostredí konštruujeme prvok List tak, že presúvame ikony jednotlivých stĺpcov na správne miesto

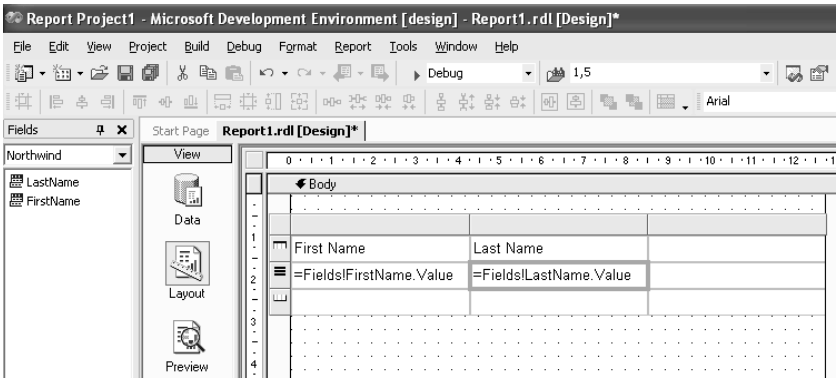


Obr. 3.13 Prvok List v návrhovom zobrazení

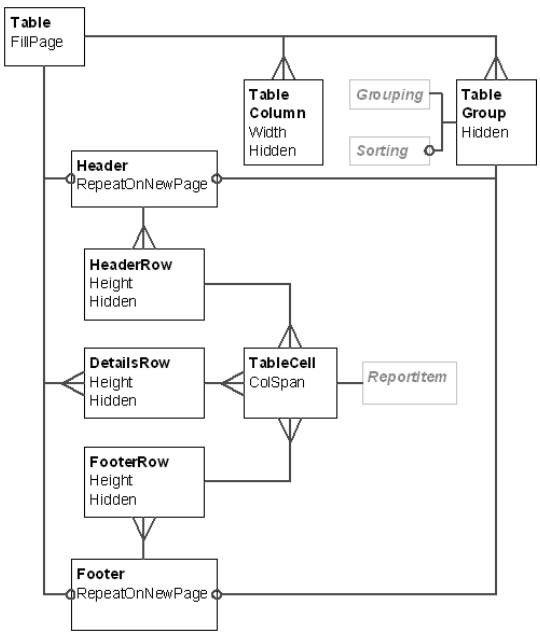
```
<ReportItems>
      <List Name="list1">
        <ReportItems>
          <Textbox Name="FirstName">
            <ZIndex>1</ZIndex>
            <Style>
              <PaddingLeft>2pt</PaddingLeft>
              <PaddingBottom>2pt</PaddingBottom>
              <PaddingTop>2pt</PaddingTop>
              <PaddingRight>2pt</PaddingRight>
            </Style>
            <rd:DefaultName>FirstName</rd:DefaultName>
            <Height>0.25in</Height>
            <Width>1.625in</Width>
            <Top>0.3174603cm</Top>
            <Value>=Fields!FirstName.Value</Value>
            <Left>0.3174603cm</Left>
          </Textbox>
          <Textbox Name="LastName">
            ...
          </Textbox>
        </ReportItems>
        <Height>1.125in</Height>
        <Width>4.375in_u47 ?Width>
        <DataSetName>DataSet1</DataSetName>
        <Top>0.3174603cm</Top>
        <Left>0.3174603cm</Left>
      </List>
    </ReportItems>
```

Table

Tabuľka je zrejme najbežnejšou formou pre zobrazovanie údajov z databáz. Tabuľka je organizovaná do riadkov a stĺpcov, pričom údaje sa nachádzajú v poliach tabuľky, ktoré sú v priesečníkoch riadkov a stĺpcov. Základnou vlastnosťou tabuľky je konštatná šírka každého stĺpca, stĺpce sú samozrejme pevné. Vždy je možné zmeniť len šírku stĺpca ako celku, nikdy sa nedá nastaviť rôzna šírka buniek v tom istom stĺpci. Údaje v stĺpcoch sa ale môžu prekrývať. Údaje v bunkách tabuľky môžu byť rôzne zarovnané, napríklad text naľavo, číselné hodnoty napravo a podobne. Tabuľka však môže obsahovať niektoré riadky, ktoré sú mimo stĺpcovej organizácie, napríklad pre rôzne súčty a medzisúčty. Pri vizuálnom návrhu tabuľky do prvého riadku návrhového zobrazenia tabuľky umiestnime názvy stĺpcov, do druhého riadku názvy stĺpcov databázovej tabuľky, ktorých hodnoty budeme chcieť zobraziť a do posledného riadku prípadné medzisúčty a podobne



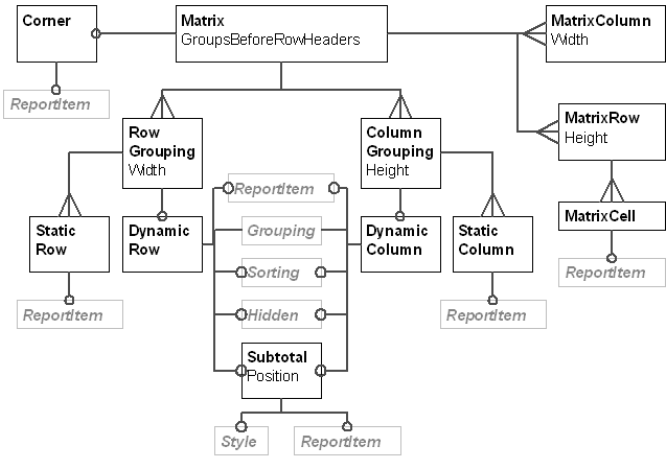
Obr. 3.14 Vizuálny návrh tabuľky



Obr. 3.15 Prvok tabuľka v schéme jazyka RDL

Matrix

Matice sú podobné tabuľkám, s tým rozdielom, že opakovať sa môžu riadky a stĺpce. Známe sú tiež pod názvom kontingenčná tabuľka (pivot table), prípadne krížová tabuľka (cross table). Medzisúčty môžu byť formátované odlišne.



Obr. 3.16 Prvok Matrix v schéme jazyka RDL

Chart

Pre zobrazovanie grafov sa používa komponenta Dundas Charts.

Výrazy v jazyku RDL

Výrazy v jazyku RDL používajú syntax programovacieho jazyka VisualBasic.NET. Hodnoty môžu byť položky z data setu, napríklad názvy stĺpcov, hodnoty iných entít z reportov, hodnoty globálnych premenných a parametrov, agregačné funkcie (Sum, Avg, Count, Min, First...) a podobne

napríklad

```
=Fields!Name.Value
=Fields!First.Value & " " & Fields!Last .Value
=Sum(Fields!Sales.Value)
=IIF(Sum(Fields!Sales.Value)>10, "green", "red")
```

Tento zápis je pochopiteľne potrebné objasniť. Pred výkričníkom sa nachádzajú názvy globálnych objektov.

Collection	Popis	Dátový typ
Fields	Fields aktuálneho data setu	*Field*
Parameters	Parametre reportu	*Parameter*
ReportItems	Textboxy reportu	*ReportItem*
Globals	Globálne premenné	*Variant*
User	Údaje špecifikované používateľom	Variant

syntax podľa pravidiel Visual Basicu .NET potom bude

Collection!ObjectName alebo ***Collection.Item(„ObjectName“)*** prípadne *Collection(„ObjectName“)*
Napríklad: *User!Language*
môžeme použiť aj syntax:
Collection.ObjectName
Napríklad: *Globals.PageNumber*

Datové typy

RDL Typ	CLR Typy
String	String, Char
Boolean	Boolean
Integer	Int16, Int32, Int64, Uint16, Uint32, Uint64, Byte, Sbyte, Timespan
DateTime	DateTime
Float	Single, Double, Decimal
Binary	Byte[
Variant	Any of the above except Byte[
VariantArray	Array of Variant

Agregačné funkcie

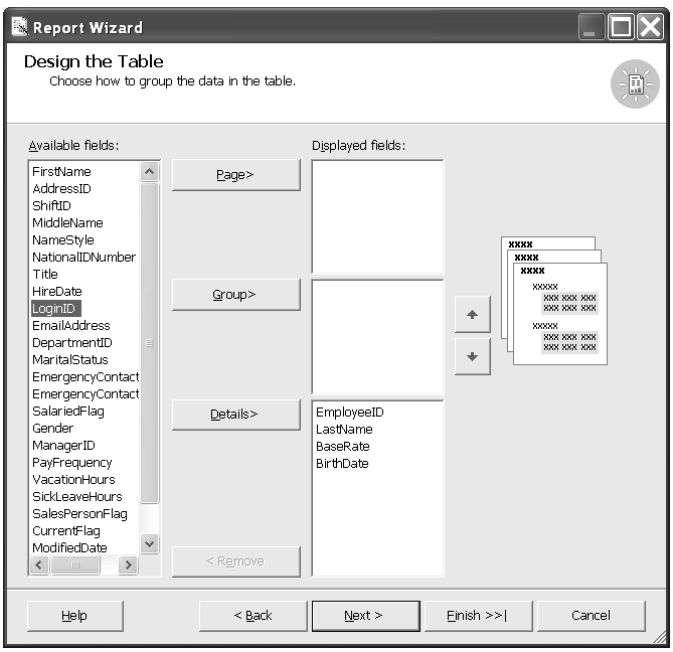
Agregačné funkcie operujú nad množinou záznamov (riadkov databázovej tabuľky), pričom vracajú jeden výsledok pre celú vstupnú množinu údajov. Z uvedeného vyplýva, že tieto funkcie pomocou matematických a štatistických operácií spracovávajú agregované hodnoty v celých stĺpcoch.

Skôr než sa dostaneme k popisu jednotlivých agregačných funkcii stojí za zmienku ukázať vytvorenie jednoduchého projektu kde budeme môcť s agregačnými funkciami v prípade záujmu experimentovať. Použijeme sprievodcu pre vytvorenie reportu. Ako databázu zadáme AdventureWorks2000, takže pripojovací reťazec (v závislosti od názvu SQL Servera) bude v tvare

```
data source=NCLL;initial catalog=AdventureWorks2000
```

Budú to jednoduché experimenty, takže využijeme tabuľku zamestnancov Employee. Pre vytvorenie SQL dopytu nepotrebujeme sprievodcu, zadáme ho priamo

```
SELECT * FROM Employee
```

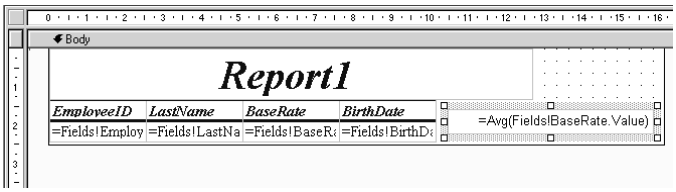


Obr. 3.17 Návrh reportu pre precvičenie agregačných funkcii

Report bude typu Table.
V záverečnom dialógu by sme potom mali mať zobrazené tieto „parametre“ projektu

```
Data Source:          AdventureWorks2000
Connection String:    data source=LL2K3;initial catalog=AdventureWorks2000
Report Type:          Table
Layout Type:          Stepped
Style:                Corporate
Details:              EmployeeID, LastName, BaseRate, BirthDate
Query:                SELECT * FROM Employee
```

K základnému návrhu reportu pridáme jeden textbox, do ktorého môžeme umiestňovať agregačné funkcie.



Obr. 3.18 Projekt pre precvičenie agregačných funkcii

Pri ich vytváraní si môžeme pomôcť jednoduchým trikom. Ak z ľavej časti pracovnej obrazovky, z poľa Fields presunieme ikonku niektorého stĺpca, ktorý je možné agregovať do textboxu, automaticky bude do textboxu umiestnená agregačná funkcia SUM. A túto stačí prepísať...

Pomocou parametra SCOPE prípadne môžeme určiť na aké údaje je funkcia zameraná, a podla coho sú údaje zoskupené, napríklad Count(*, "Orders")

Avg
Funkcia vráti aritmetický priemer z množiny hodnôt. Z matematického hľadiska je to súčet hodnôt vydelený ich počtom.
Syntax: *Avg(Expression, Scope)*
Príklad: =Avg(Fields!BaseRate.Value)

Count
Funkcia vráti počet hodnôt v množine údajov.
Syntax: Count(Expression, Scope)
Count(*, Scope)

Príklad: =Count(*)
=Count(Fields!EmployeeID.Value)

CountDistinct
Funkcia vráti počet unikátnych hodnôt (hodnota sa započíta len raz) v množine údajov.
Syntax: *CountDistinct(Expression, Scope)*
Príklad: =CountDistinct(Fields!EmployeeID.Value)

First
Vráti prvú hodnotu definovanú výrazom
Syntax: *First(Expression, Scope)*
Príklad: =First(Fields!BaseRate.Value)

Last
Vráti poslednú hodnotu definovanú výrazom
Syntax: *Last(Expression, Scope)*
Príklad: =Last(Fields!BaseRate.Value)

Max
Funkcia vráti maximálnu hodnotu z množiny údajov. Hodnoty NULL sa ignorujú.
Syntax: *Max(Expression, Scope)*
Príklad: =Max(Fields!BaseRate.Value)

Min
Funkcia vráti minimálnu hodnotu z množiny údajov. Hodnoty NULL sa ignorujú.
Syntax: *Min(Expression, Scope)*
Príklad: =Min(Fields!BaseRate.Value)

RowNumber
Počet riadkov v zoskupení.
Syntax: *RowNumber(Scope)*

RunningValue
Vráti bežiace agregácie pre výraz

Syntax: *RunningValue(Expression, Function, Scope)*
Príklad: RunningValue(Fields!Cost.Value, Sum, Nothing)

StDev, StDevP
Funkcia vráti štandardnú odchýlku (odmocninu z variancie) pre skupinu záznamov.
Syntax: *StDev(Expression, Scope)*

Sum
Funkcia vráti súčet hodnôt v množine údajov špecifikovanej výrazom.
Syntax: *Sum(Expression, Scope)*
Príklad: =Sum(Fields!BaseRate.Value)

Var, VarP
Funkcia vráti varianciu (populačnú) pre skupinu záznamov.

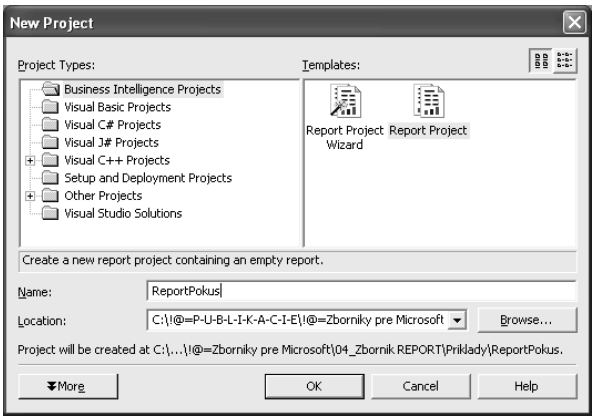
Syntax: *Var(Expression, Scope)*

Report Designer – prvý projekt
Po naznačení oblasti a spôsobu použitia reportovacích služieb SQL Servera môžeme pristúpiť k prvému praktickému sa zoznámeniu s produktom. Mnohým čitateľom možno jednoduchá ukážka napovie viac, než rozsiahla teória. Návrh reportu pomocou Report Designera bude prebiehať vo vývojovom prostredí Visual Studio.NET 2003. Aby sme vylúčili vplyv predchádzajúcich nastavení a pri tvorbe reportov mali zobrazené všetky okná, ešte pred začiatkom vytvárania projektu môžeme pomocou položky View hlavného menu nechať zobraziť menu

- Solution Explorer
- Properties Window
- Toolbox

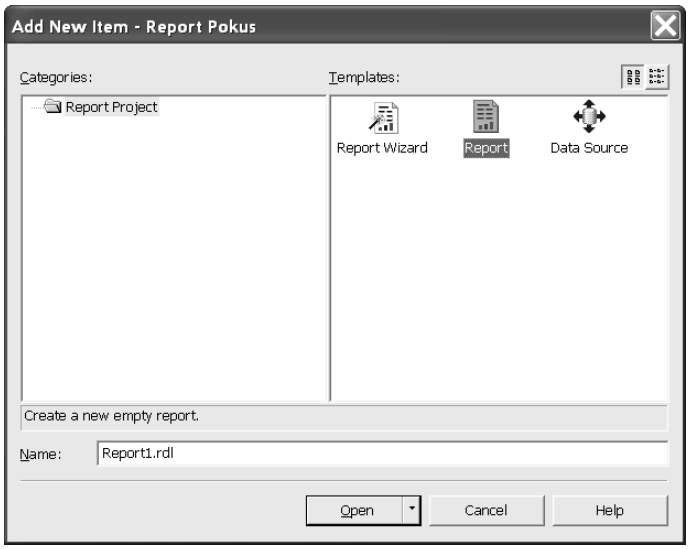
Teraz už môžeme pristúpiť k vytvoreniu projektu. V hlavnom menu aktivujeme položku menu pre vytvorenie nového projektu. V zložke typov projektov Business Intelligence Projects máme k dispozícii dve šablóny projektov
- Report Project Wizard
- Report Project

Použijeme šablónu Report Project a nový projekt nazveme napríklad ReportPokus.



Obr. 3.19 Dialóg pre vytvorenie nového projektu vo vývojovom prostredí Visual Studio .NET 2003

Po vytvorení projektu návrhu reportu ako prvý krok vytvoríme RDL súbor, ktorý bude predpis pre návrh reportu obsahovať. V okne Solution Explorer v pravej časti pracovnej obrazovky vývojového prostredia vyberieme zložku Reports. Pre túto zložku aktivujeme kontextové menu „Add new Item“ a v ňom šablónu Report.

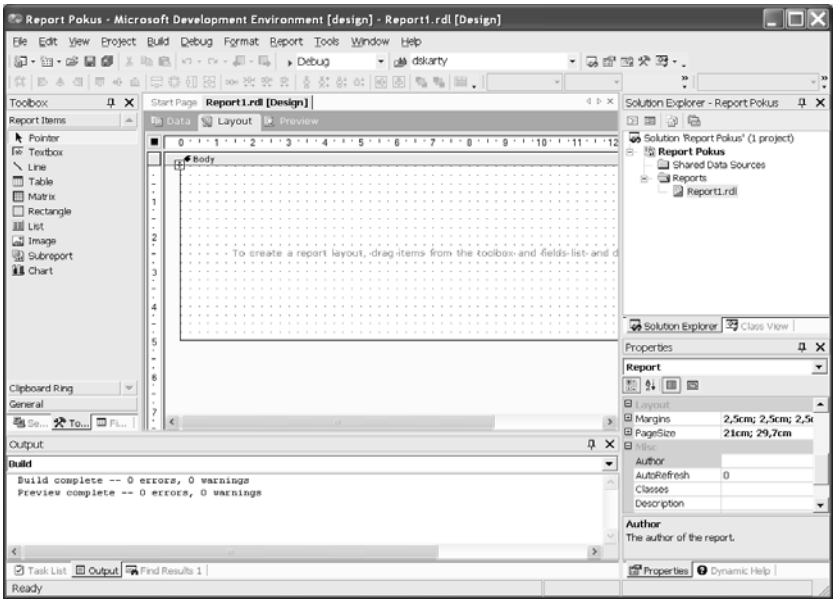


Obr. 3.20 Dialóg pre vytvorenie nového reportu

Po vytvorení nového projektu sa môžeme zoznámiť z pracovnou obrazovkou vývojového prostredia v režime Report Designer. Ľavé okno Toolboxu budeme využívať spravidla v dvoch režimoch (záložkách). V režime Report Items pre vizuálny návrh reportu pomocou komponentov a v režime Fields pre prácu s databázovými objektami. Nasleduje okno pre prepínanie hlavnej pracovnej ploche. Túto plochu je možné prepnúť do režimov

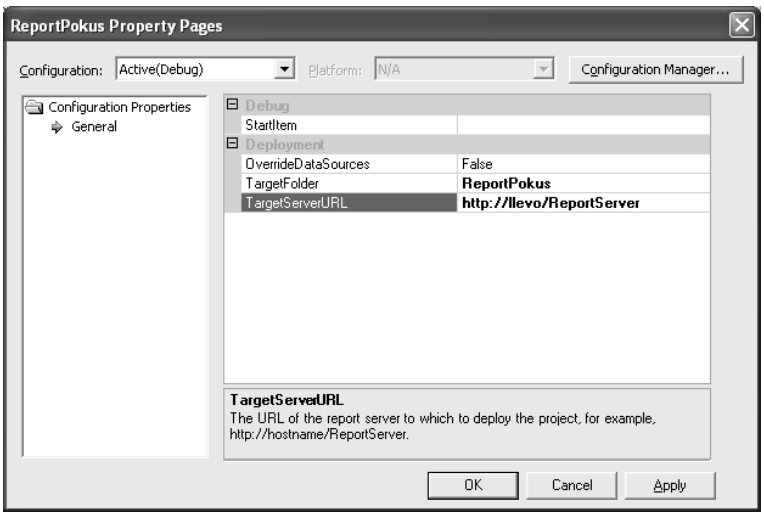
– **Data** pre pripojenie sa k databáze a návrh SQL dopytu
– **Layout** pre návrh formulára reportu
– **Preview** pre prehliadanie reportu.

Nasleduje hlavné pracovné okno aplikácie a úplne vpravo sú pod sebou okná Solution Explorer a Properties.



Obr. 3.21 Report Designer vo vývojovom prostredí Visual Studio .NET 2003 – pracovná plocha vývojového prostredia po vytvorení nového projektu

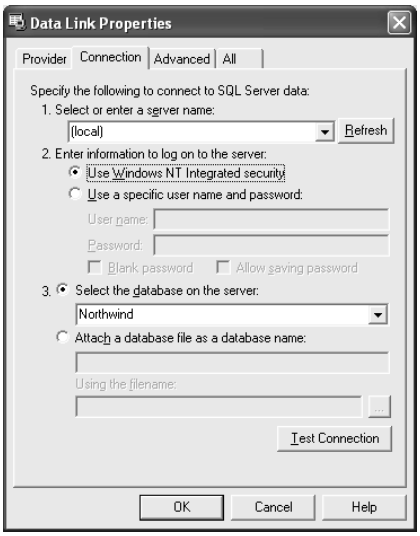
Jedno z prvých nastavení, ktoré v novom projekte urobíme, je nastavenie adresára projektu a URL adresy reportovacieho servera. V okne Solution Explorer pravým tlačidlom myši aktivujeme kontextové menu na položku ReportPokus (názov projektu). Vyberieme položku Properties. Zobrazí sa dialóg ReportPokus, kde zadáme obidva spomínané parametre.



Obr. 3.22 Nastavenie cieľového adresára a URL adresy report servera.

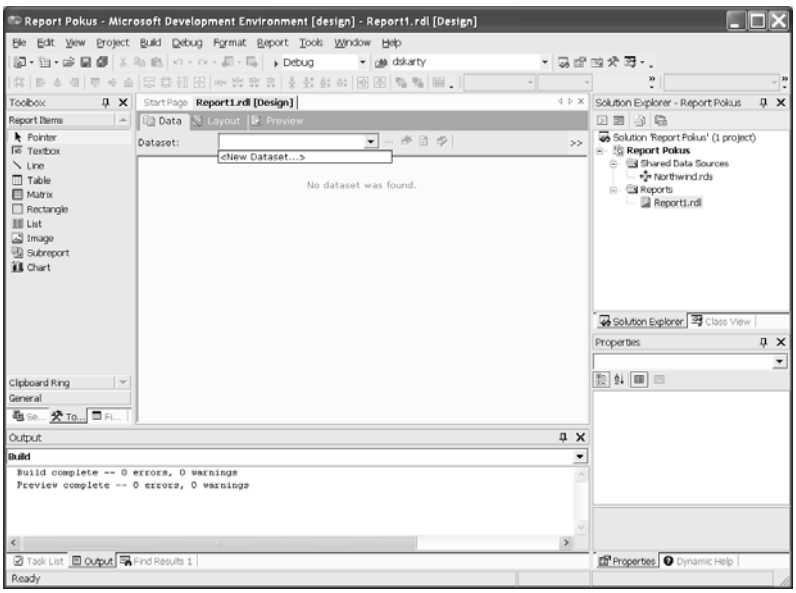
Výber údajov – návrh SQL dopytu

Po založení je potrebné vytvoriť prepojenie na zdroj údajov. Po aktivácii kontextového menu na zložku Shared Data Source vyberieme zdroj údajov. Pre náš prípad sa pripojíme k cvičnej databáze Northwind, ktorá sa nainštaluje spolu s SQL Serverom 2000.



Obr. 3.23 Pripojenie sa k zdroju údajov

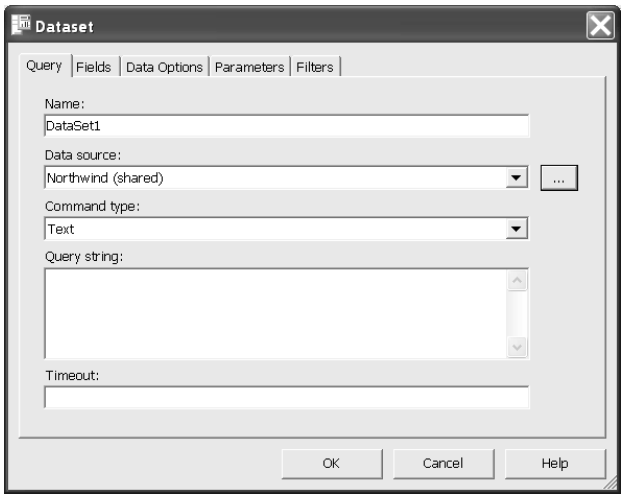
Obrazovku vývojového prostredia nastavíme do režimu návrhu SQL dopytu pre výber údajov. V okne View je vtedy označená aktívna záložka Data.



Obr. 3.24 Pracovná plocha vývojového prostredia po vytvorení nového projektu a pripojení sa k zdroju údajov

Pre prípadné priame SQL dopytu klikneme na tlačidlo označené tromi bodkami (...). Zobrazí sa dialóg pre vytvorenie DataSetu. Dialóg má päť záložiek

- Query
- Fields
- Parameters
- Data Options
- Filters

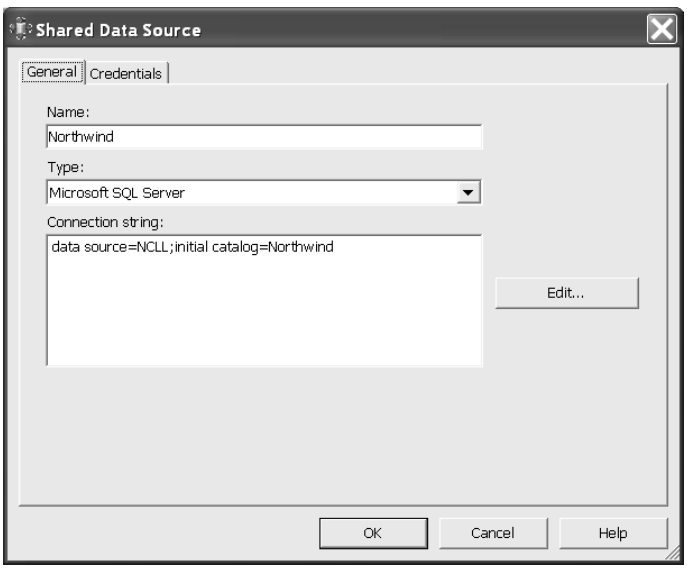


Obr. 3.25 Dialóg pre vytvorenie DataSetu

Zdroj údajov (databázu Northwind) sme už špecifikovali v procese vytvorenia nového zdroja údajov
Do editačného poľa Query string zadáme SQL dopyt pre výber údajov, napríklad

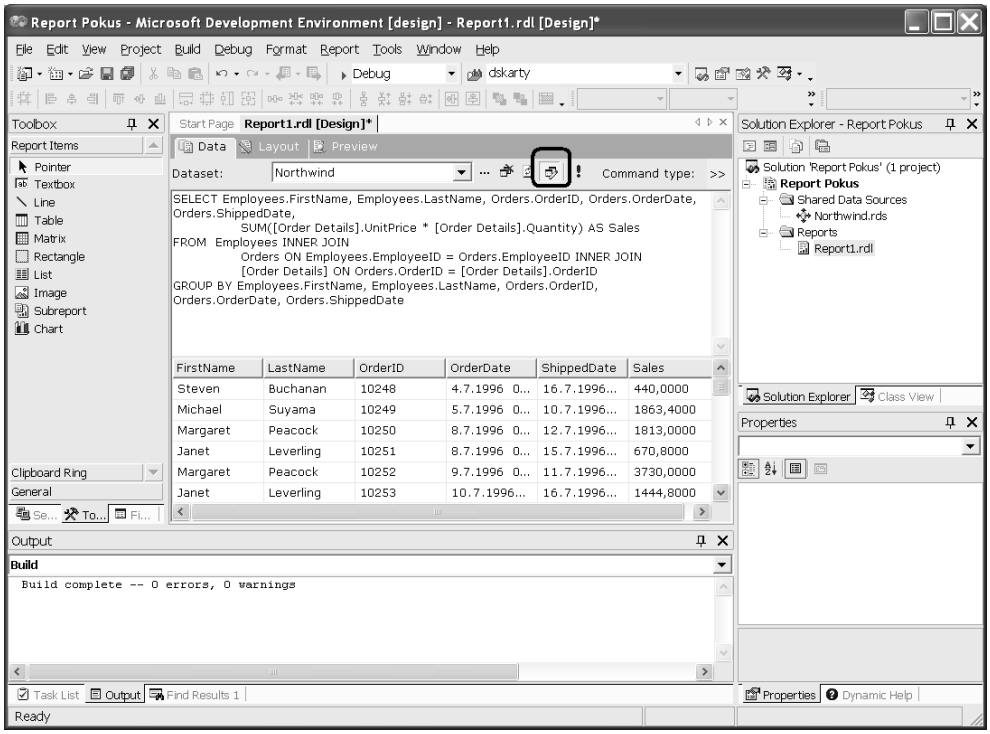
```
SELECT    Employees.FirstName, Employees.LastName, Orders.OrderID, Orders.OrderDate,
         Orders.ShippedDate, SUM([Order Details].UnitPrice * [Order Details].Quantity) AS Sales
FROM          Employees INNER JOIN
                 Orders ON Employees.EmployeeID = Orders.EmployeeID INNER JOIN
                 [Order Details] ON Orders.OrderID = [Order Details].OrderID
GROUP BY  Employees.FirstName, Employees.LastName, Orders.OrderID, Orders.OrderDate,
          Orders.ShippedDate
```

Kliknutím na tlačidlo označené tromi bodkami (...) vedľa položky Data source môžeme prípadne zobraziť dialóg pre zistenie, prípadne nastavenie parametrov zdroja údajov. Dôležitý je hlavne reťazec obsahujúci parametre pre pripojenie sa k zdroju údajov



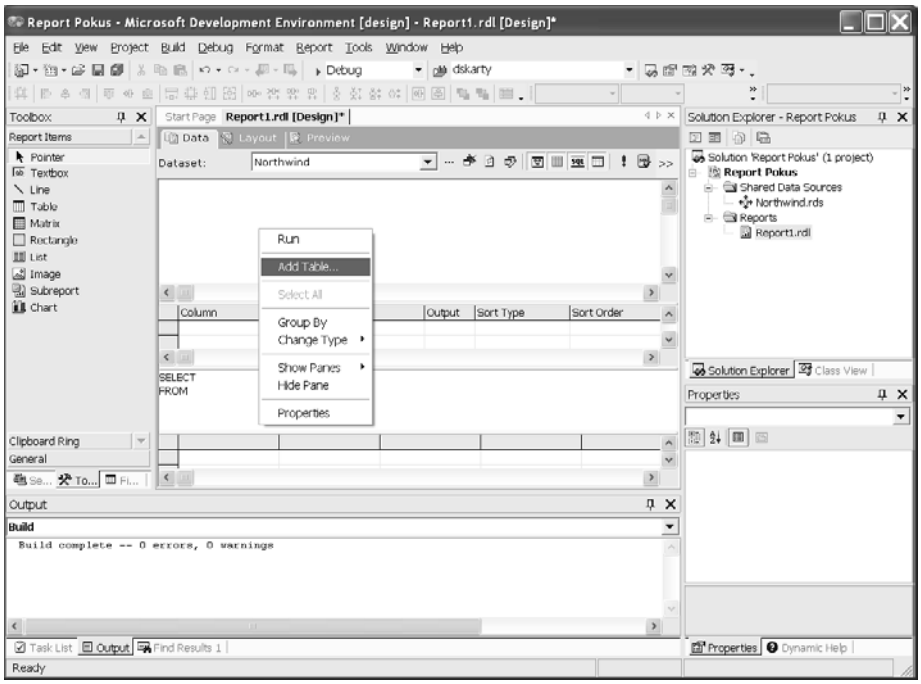
Obr. 3.26 Dialóg pre nastavenie parametrov pre pripojenie sa k zdroju údajov

Vráťme sa však na pracovnú plochu vývojového prostredia. V strednom okne bude po výbere datasetu zobrazená analógia konzoly databázového servera. V hornej polovici je zadaný SQL dopyt a v dolnej polovici ukážka údajov.



Obr. 3.27 Nástroj pre zadávanie a testovanie SQL dopytu

Pomocou tlačidla Generic Query Designer je možné prepnúť hlavú časť obrazovky do režimu vizuálneho interaktívneho budovania SQL dopytu. V kontextovom menu aktivovanom na ploche označíme položku Add Table



Obr. 3.28 Grafický návrh SQL dopytu

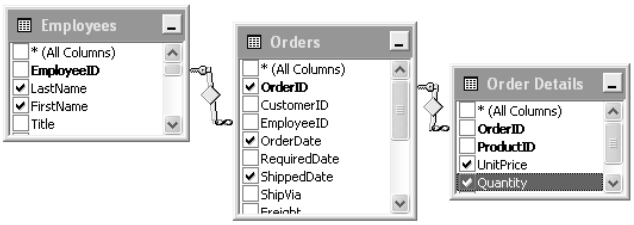
Zo zoznamu tabuliek databázy Northwind vyberieme tabuľky
- Employees
- Orders
- Order Details

Ak sledujeme ako sa konštruuje SQL dopyt, tak v tejto etape výberu tabuliek sa vytvorila časť SQL dopytu za klauzulou FROM, kde sú jednotlivé tabuľky zviazané pomocou klauzule INNER JOIN

```
SELECT
FROM  Employees INNER JOIN
                Orders ON Employees.EmployeeID = Orders.EmployeeID INNER JOIN
                [Order Details] ON Orders.OrderID = [Order Details].OrderID
```

Pomocou nástroja Query Builder zahrnieme do dopytu (zaškrtnutím check boxov) tieto stĺpce predmetných tabuliek.

Tabuľka	Stĺpec
Employee	FirstName
Employee	LastName
Orders	OrderID
Orders	OrderDate
Orders	ShippedDate
Order Details	UnitPrice
Order Details	Quantity



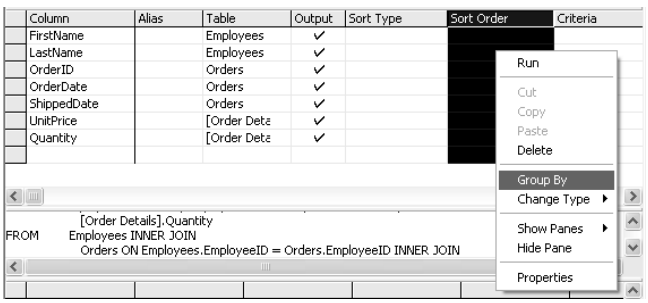
Obr. 3.29 Výber stĺpcov pre SQL dopyt prostredníctvom nástroja Query Builder

Výberom konkrétnych stĺpcov konkrétnych tabuliek v návrhovom zobrazení Query Buildera bola skonštruovaná úvodná časť SQL dopytu, hneď za klauzulou SELECT

Pomocou dosiaľ vykonaných krokov n návrhovom zobrazení sme už vytvorili značnú časť SQL dopytu, zostáva nám ešte vytvoriť klauzulu GROUP BY pre zoskupenie výsledkov

```
SELECT  Employees.FirstName, Employees.LastName, Orders.OrderID, Orders.OrderDate,
        Orders.ShippedDate, [Order Details].UnitPrice, [Order Details].Quantity
FROM    Employees INNER JOIN
            Orders ON Employees.EmployeeID = Orders.EmployeeID INNER JOIN
            [Order Details] ON Orders.OrderID = [Order Details].OrderID
```

Klauzulu GROUP BY vytvoríme nasledovným spôsobom. Kliknutím na záhlavie stĺpca Sort Order označíme tento stĺpec a v kontextovom menu aktivujeme položku GROUP BY



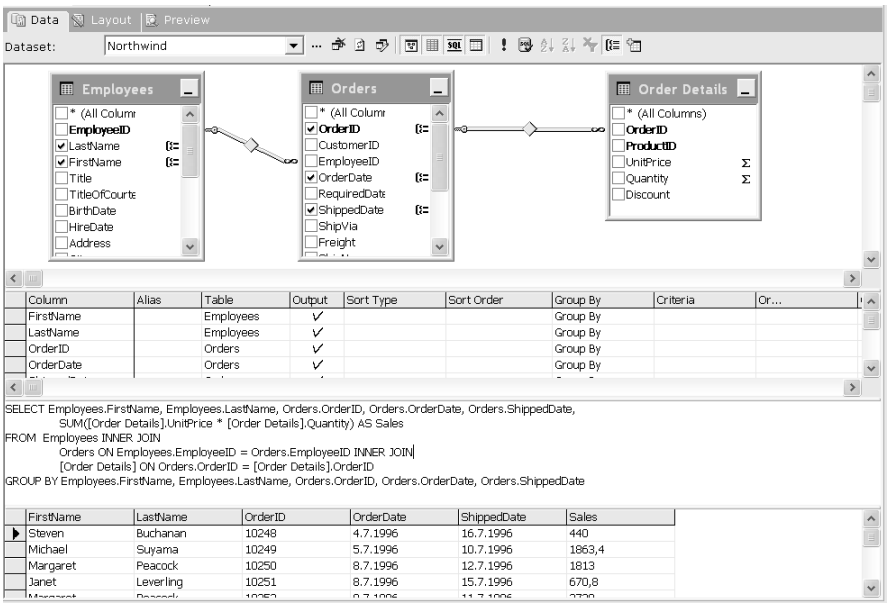
Obr. 3.30 Návrh klauzule Group By prostredníctvom nástroja Query Builder

Týmto krokom sa do klauzuly GROUP BY zahrnú všetky stĺpce. Z tohoto výberu odstránime stĺpce UnitPrice
Z Klauzuly SELECT odstránime priamo v SQL dopyte stĺpce "[Order Details].UnitPrice, [Order Details].Quantity
a nahradíme ich agregačnou funkciou

SUM([Order Details].UnitPrice * [Order Details].Quantity) AS Sales

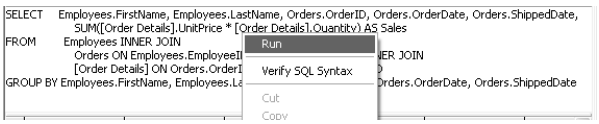
Výsledný SQL dopyt bude potom

```
SELECT   Employees.FirstName, Employees.LastName, Orders.OrderID, Orders.OrderDate,
         Orders.ShippedDate, SUM([Order Details].UnitPrice * [Order Details].Quantity) AS Sales
FROM          Employees INNER JOIN
                 Orders ON Employees.EmployeeID = Orders.EmployeeID INNER JOIN
                 [Order Details] ON Orders.OrderID = [Order Details].OrderID
GROUP BY  Employees.FirstName, Employees.LastName, Orders.OrderID, Orders.OrderDate,
          Orders.ShippedDate
```



Obr. 3.31 Kompletne zostavený SQL dopyt

Zostavený SQL dopyt môžeme vyskúšať, či je po spomínaných úpravách naozaj syntakticky správny, či v ňom napríklad nechýba nejaká čiarka a podobne. Dopyt môžeme vyskúšať buď priamo vo vývojovom prostredí pomocou kontextového menu (položka Run) aktivovanom v okne, kde sa nachádza text dopytu



Obr. 3.32 Otestovanie SQL dopytu

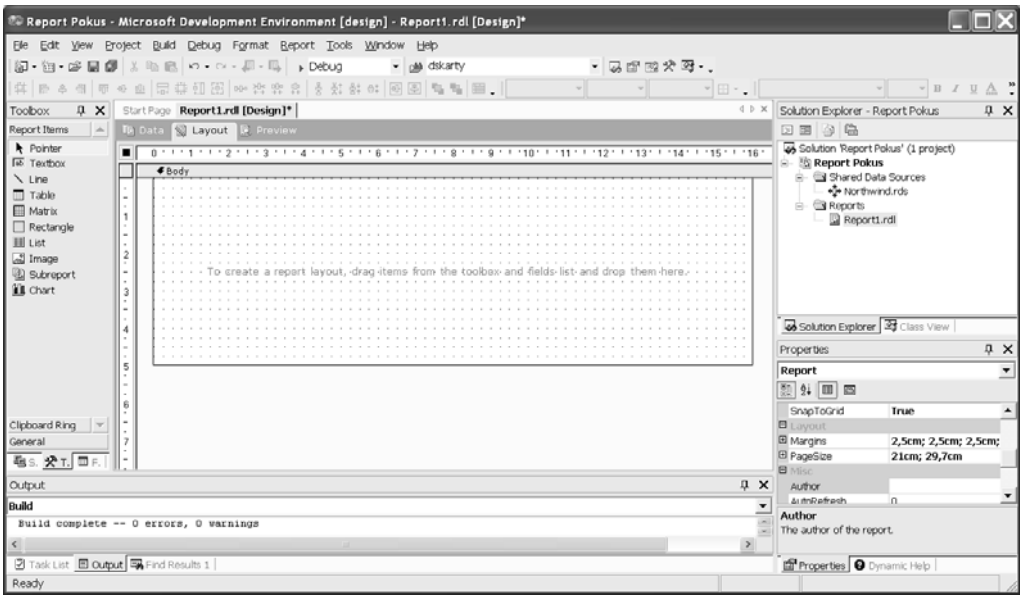
Dopyt môžeme vyskúšať aj v konzolovej aplikácii Query Analyzer do ktorej text SQL dopytu prekopírujeme a spustíme. Výsledok dopytu bude

```
FirstName  LastName             OrderID     OrderDate    ShippedDate  Sales
---------- -------------------- ----------- -----------  -----------  ---------------------
Steven     Buchanan             10248       1996-07-04   1996-07-16   440.0000
Michael    Suyama               10249       1996-07-05   1996-07-10   1863.4000
Margaret   Peacock              10250       1996-07-08   1996-07-12   1813.0000
Janet      Leverling            10251       1996-07-08   1996-07-15   670.8000
Margaret   Peacock              10252       1996-07-09   1996-07-11   3730.0000
Janet      Leverling            10253       1996-07-10   1996-07-16   1444.8000
```

```
Steven      Buchanan            10254       1996-07-11  1996-07-23  625.2000
Anne        Dodsworth           10255       1996-07-12  1996-07-15  2490.5000
Janet       Leverling           10256       1996-07-15  1996-07-17  517.8000
Margaret    Peacock             10257       1996-07-16  1996-07-22  1119.9000
Nancy       Davolio             10258       1996-07-17  1996-07-23  2018.6000
...
(830 row(s) affected)
```

Návrh reportu

Po ukončení návrhu SQL dopytu a jeho otestovaní môžeme pristúpiť k návrhu reportu. Vývojové prostredie prepneme do režimu Layout.



Obr. 3.33 Visual Studio .NET 2003 v režime návrhu reportu

V záložke návrhového okna vidíme názov súboru reportu. Súbor Report1.rdl ktorý obsahuje všetky údaje o vytváranom reporte je klasický XML dokument. Po vytvorení nového projektu obsahuje údaje:

```
<?xml version="1.0" encoding="utf-8"?>
<Report xmlns="http://schemas.microsoft.com/sqlserver/reporting/2003/10/reportdefinition"
xmlns:rd="http://schemas.microsoft.com/SQLServer/reporting/reportdesigner">
       <Width>6.5in</Width>
       <Body>
              <Height>2in</Height>
       </Body>
       <rd:InitialLanguage>true</rd:InitialLanguage>
       <rd:InitialDimensions>
              <rd:UnitType>Inch</rd:UnitType>
              <rd:LeftMargin>1in</rd:LeftMargin>
              <rd:RightMargin>1in</rd:RightMargin>
              <rd:TopMargin>1in</rd:TopMargin>
              <rd:BottomMargin>1in</rd:BottomMargin>
              <rd:PageWidth>8.5in</rd:PageWidth>
              <rd:PageHeight>11in</rd:PageHeight>
              <rd:ColumnSpacing>0.5in</rd:ColumnSpacing>
       </rd:InitialDimensions>
       <rd:InitialDimensions>
              <rd:UnitType>Cm</rd:UnitType>
              <rd:Width>16cm</rd:Width>
              <rd:Height>5cm</rd:Height>
```

```
            <rd:LeftMargin>2.5cm</rd:LeftMargin>
            <rd:RightMargin>2.5cm</rd:RightMargin>
            <rd:TopMargin>2.5cm</rd:TopMargin>
            <rd:BottomMargin>2.5cm</rd:BottomMargin>
            <rd:GridSpacing>0.25cm</rd:GridSpacing>
            <rd:PageWidth>21cm</rd:PageWidth>
            <rd:PageHeight>29.7cm</rd:PageHeight>
            <rd:ColumnSpacing>1cm</rd:ColumnSpacing>
      </rd:InitialDimensions>
</Report>
```

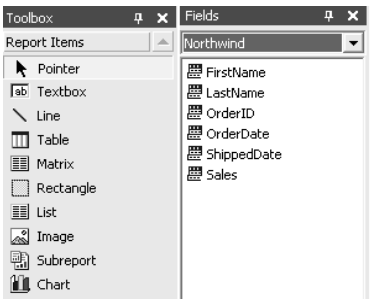
Po definovaní zdroja údajov bude zatiaľ ešte stále prázdny report tvorený kódom

```
<?xml version="1.0" encoding="utf-8"?>
<Report xmlns="http://schemas.microsoft.com/sqlserver/reporting/2003/10/reportdefinition"
xmlns:rd="http://schemas.microsoft.com/SQLServer/reporting/reportdesigner">
  <rd:GridSpacing>0.25cm</rd:GridSpacing>
  <RightMargin>2.5cm</RightMargin>
  <Body>
    <Style />
    <Height>5cm</Height>
    <ColumnSpacing>1cm</ColumnSpacing>
  </Body>
  <TopMargin>2.5cm</TopMargin>
  <DataSources>
    <DataSource Name="Northwind">
      <rd:DataSourceID>749613d4-5965-4004-97f7-48897d8ef561</rd:DataSourceID>
      <DataSourceReference>Northwind</DataSourceReference>
    </DataSource>
  </DataSources>
  <Width>16cm</Width>
  <DataSets>
    <DataSet Name="Northwind">
      <Fields>
        <Field Name="FirstName">
          <DataField>FirstName</DataField>
       <rd:TypeName>System.String</rd:TypeName>
        </Field>
        <Field Name="LastName">
          <DataField>LastName</DataField>
          <rd:TypeName>System.String</rd:TypeName>
        </Field>
        <Field Name="OrderID">
          <DataField>OrderID</DataField>
          <rd:TypeName>System.Int32</rd:TypeName>
        </Field>
        <Field Name="OrderDate">
          <DataField>OrderDate</DataField>
          <rd:TypeName>System.DateTime</rd:TypeName>
        </Field>
        <Field Name="ShippedDate">
          <DataField>ShippedDate</DataField>
          <rd:TypeName>System.DateTime</rd:TypeName>
        </Field>
        <Field Name="Sales">
          <DataField>Sales</DataField>
          <rd:TypeName>System.Decimal</rd:TypeName>
    </Field>
      </Fields>
```

```
      <Query>
        <DataSourceName>Northwind</DataSourceName>
        <CommandText>SELECT Employees.FirstName, Employees.LastName, Orders.OrderID,
Orders.OrderDate, Orders.ShippedDate,
                SUM([Order Details].UnitPrice * [Order Details].Quantity) AS Sales
FROM  Employees INNER JOIN
                Orders ON Employees.EmployeeID = Orders.EmployeeID INNER JOIN
                [Order Details] ON Orders.OrderID = [Order Details].OrderID
GROUP BY Employees.FirstName, Employees.LastName, Orders.OrderID,
Orders.OrderDate, Orders.ShippedDate</CommandText>
      </Query>
    </DataSet>
  </DataSets>
  <LeftMargin>2.5cm</LeftMargin>
  <rd:SnapToGrid>true</rd:SnapToGrid>
  <PageHeight>29.7cm</PageHeight>
  <rd:DrawGrid>true</rd:DrawGrid>
  <PageWidth>21cm</PageWidth>
  <rd:ReportID>5da7806e-c63e-4b52-aaea-362a3816a491</rd:ReportID>
  <BottomMargin>2.5cm</BottomMargin>
  <Language>en-US</Language>
</Report>
```

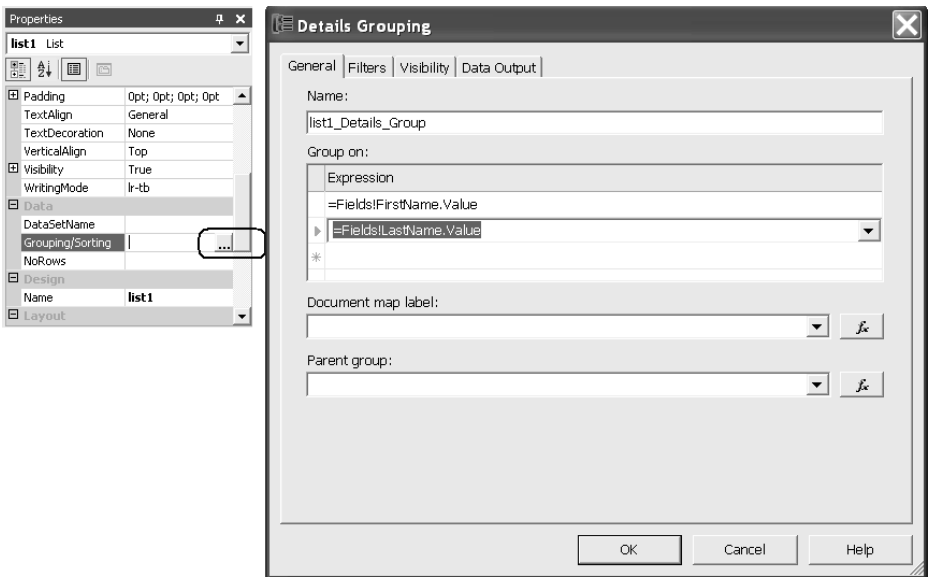
Pre ľahšiu orientáciu sú začiatočné a koncové tagy elementov vyznačené hrubám fontom

Ľavé okno vývojového prostredia je možné pomocou záložiek prepnúť do režimu Toolbox, prípadne Fields.



Obr. 3.34 Visual Studio .NET 2003 – záložky Toolbox a Fields

Návrh formulára zahájime presunom komponenty **List** z okna toolboxu na plochu reportu.
V okne properties pomocou tlačidla (...) aktivujeme dialóg Grouping and Sorting. V okne Grouping vyberieme polia „Fields!FirstName.Value“ a „Fields!LastName.Value“ podľa ktorých budú údaje v reportoch zoskupované.

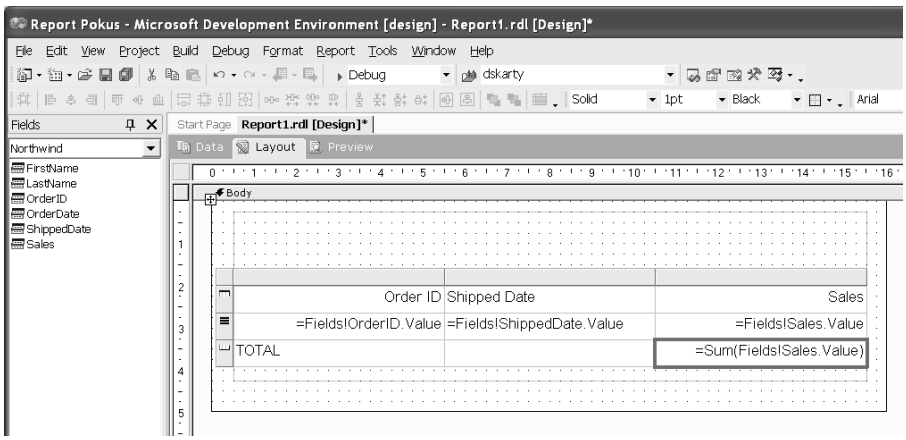


Obr. 3.35 Dialog Grouping and Sorting

Do vnútra plochy komponenty List premiestnime z Toolboxu komponentu Table. Návrhové zobrazenie Table obsahuje mriežku 3 riadky x 3 stĺpce. Prepneme ľavé okno vývojového prostredia z Toolboxu na Fields a do stredného riadku premiestnime do jednotlivých stĺpcov ikony polí

- OrderID
- ShippedDate
- Sales

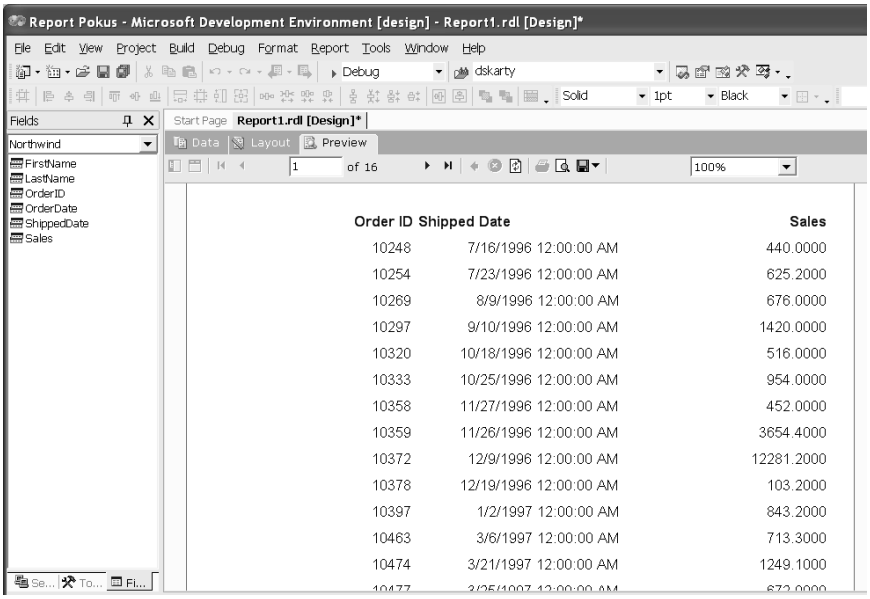
Do ľavého dolného poľa napíšeme TOTAL a do pravého dolného poľa presunieme ešte raz položku Sales. Situáciu názorne ilustruje obrázok.



Obr. 3.36 Návrhový režim komponenty Table

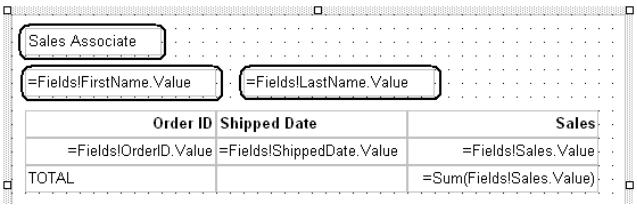
V návrhu pokračujeme naformátovaním tabuľky, konkrétne nastavením fontu. Označíme všetky tri stĺpce (stlačíme a držíme kláves CTRL a postupne označujeme všetky tri stĺpce). Následne vo formátovacom toolbare nastavíme vhodný font.

Na výsledok návrhu sa môžeme predbežne pozrieť ak prepneme hlavné okno vývojového prostredia do režimu Preview. Je tam tabuľka zatiaľ v pomerne „surovom“ stave.



Obr. 3.37 Preview reportu

Vzhľad reportu vylepšíme pridaním TextBoxov. V ľavom hornom TextBoxe bude statický text „Sales Associate“. V boxoch o riadok nižšie budú polia „FirstName“ a „LastName“.



Obr. 3.38 Pridanie TextBoxov

V tejto etape môžeme projekt uložiť príkazom Save a prehliadnuť výsledok návrhu.
Prepínaním stránok sa v Textboxoch menia mená pracovníkov

Konečným cieľom je zostavenie reportu a prenos reportu do réžie reportovacieho serveru. V menu Debug aktivujeme položku Start. V okne Output si môžeme prehliadnuť protokol o zostavení a zavedení.

```
------ Build started: Project: Report Pokus, Configuration: Debug ------

Build complete -- 0 errors, 0 warnings

------ Deploy started: Project: Report Pokus, Configuration: Debug ------

Deploying to http://ncll/ReportServer?%2fReport+Pokus
Deploying data source '/Report Pokus/Northwind'.
Deploying report 'Report1'.
Deploy complete -- 0 errors, 0 warnings



---------------------- Done ----------------------

    Build: 1 succeeded, 0 failed, 0 skipped
    Deploy: 1 succeeded, 0 failed, 0 skipped
```

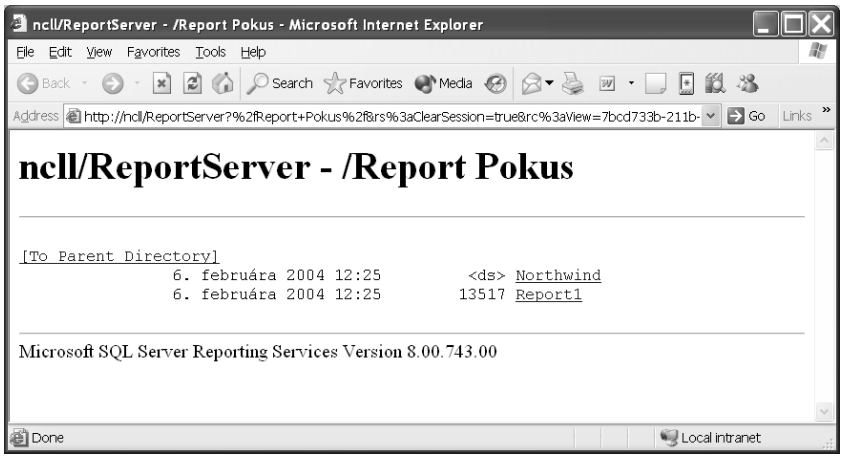
Report si môžeme prezrieť aj priamo pomocou prehliadača webového obsahu. Stačí zadať príslušnú adresu, v našom prípade `http://llevo/ReportServer/ReportPokus`

Mnohých čitateľov bude možno zaujímať kde a v akom tvare a formátoch sa vygenerovaný report na serveri nachádza, preto skôr než zadáme túto adresu prehliadaču, vysvetlíme aj túto záležitosť. Z hľadiska webového servera sa reporty nachádzajú vo virtuálnom adresári ReportServer. Ak si pozrieme parametre webového servera, zistíme že tento virtuálny adresár je (pri implicitnej inštalácii reportovacích služieb) napojený na fyzický adresár

```
C:\Program Files\Microsoft SQL Server\MSSQL\Reporting Services\ReportServer
```

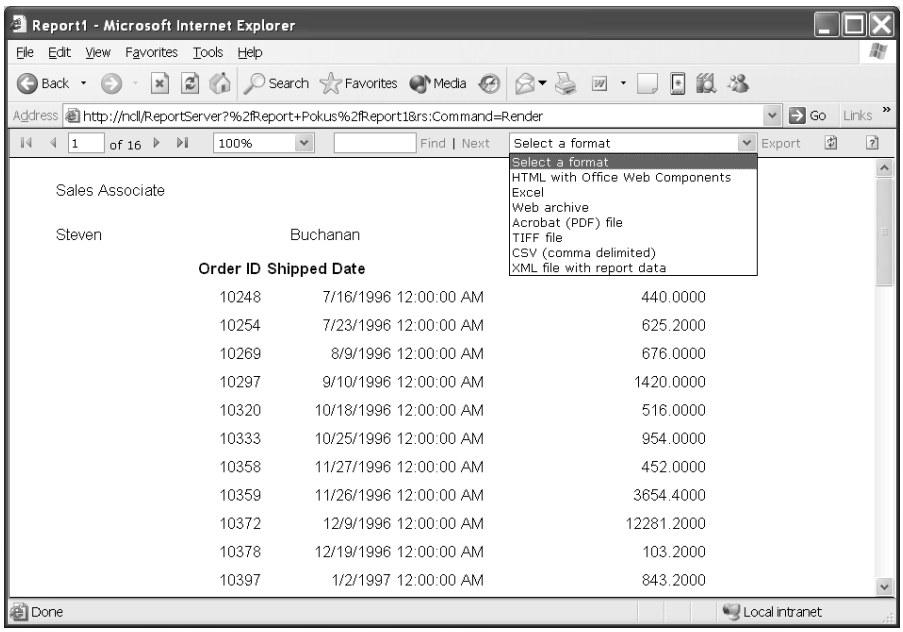
odkazy na konkrétne dokumenty z hlavnej stránky v HTML kóde sú v tvare

```
<A HREF="ReportPokus/Report1?rs:Command=Render">Report1</A>
```



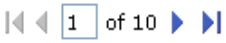
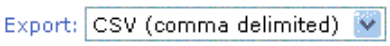
Obr. 3.39 Zobrazenie zoznamu reportov v okne HTML prehliadača

Po výbere konkrétneho reportu sa nám zobrazí v okne HTML prehliadača jeho obsah. Na obrázku je ukázané rozvinuté menu pre export.



Obr. 3.40 Zobrazenie reportu v okne HTML prehliadača

Jednotlivé symboly a ovládacie prvky reportu v okne HTML prehliadača sú vysvetlené v tabuľke

Symbol	Popis
	Navigačný prvok umožňujúci skok (symboly zľava doprava) na prvú, predošlú, konkrétne zadanú, nasledujúcu a poslednú stránku reportu. Pre prehliadanie konkrétnej strany zadáme jej číslo a potvrdíme klávesou
	Zväčšovanie alebo zmenšovanie stránky reportu v okne prehliadača. Pre prispôsobenie v horizontálnom smere môžeme zvoliť možnosť **Page Width** a pre prispôsobenie v zvislom smere **Whole Page**. Túto funkciu podporujú prehliadače počnúc verziou Internet Explorer 5.5.
	Vyhľadávanie zadaného textu v reporte (max. 255 znakov).
	Export vo zvolenom formáte a následné otvorenie okna s prehliadačom podporujúcim tento
	Zobrazenie mapy dokumentu
	Zobrazenie polí parametrov
	Aktualizácia reportu
	Nápoveda.

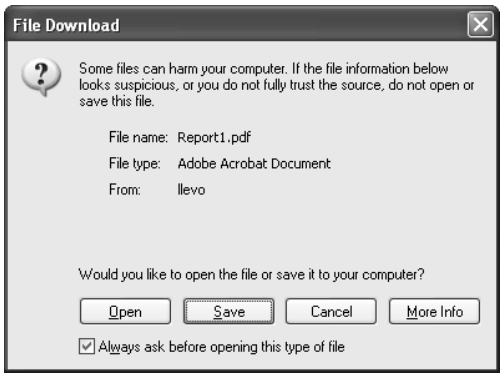
Report je možné exportovať do niekoľkých najviac používaných formátov a následne potom zobraziť v okne prehliadača. Popis podporovaných formátov pre export dokumentu je v tabuľke.

Formát exportu	Popis
Acrobat (PDF) súbor	Report vo forme PDF súboru je možné kopírovať a prenášať ako bežný súbor a taktiež je možné umiestniť ho na webový server a prehliadať pomocou prehliadačov na strane klienta, pričom vôbec nezáleží na klientskej platforme.
Webová stránka obsahujúca Office Web Components	Pre prehliadanie hlavne grafov sa použijú na klientskej strane Office Web Components. Ak nie sú k dispozícii, reportovací server vygeneruje grafy vo forme bitmáp.
CSV (comma delimited)	Súbor typu CSV, prípadne v databázovej terminológii flat súbor je použiteľný pre import do širokého spektra databázových serverov a tabuľkových procesorov, napríklad MS Excel.
Web archive	Export a prehliadanie dokumentu vo formáte MIME-encoded HTML format. Tento formát zapúzdruje v jednom dokumente aj obrázky a dokumenty na ktoré je link v hlavnom dokumente
TIFF file	Exprot do obrazového formátu TIFF. Môžeme ho prehliadať pomocou niektorého TIFF prehliadača, napríklad môžeme použiť Windows Picture and Fax Viewer. Tento formát je odporúčaný hlavne pre tlač dokumentov.

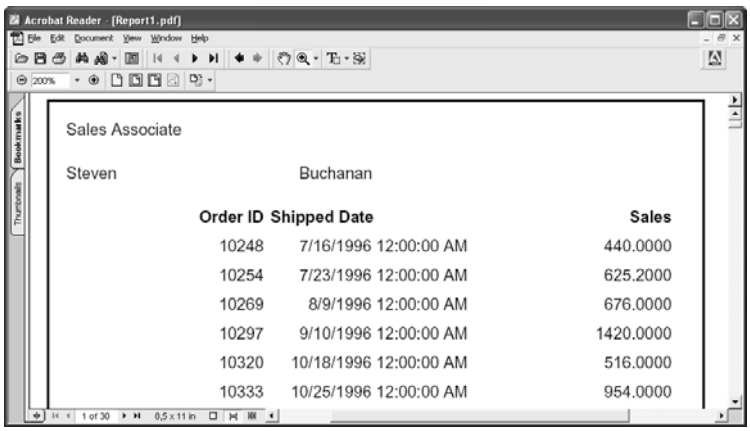
Formát CSV (Comma Separated Values) alebo po našom „hodnoty oddelené čiarkou“ je možné impotrovať do väčšiny databázových serverov, prípadne programov typu tabuľkový procesor, napríklad MS Excel.

```
textbox4,FirstName,LastName,textbox1,textbox2,textbox3,textbox7,textbox8,Sales_1,OrderID,
ShippedDate,Sales
Sales Associate,Steven,Buchanan,Order ID,Shipped
Date,Sales,TOTAL,,75567.7500,10248,7/16/1996 12:00:00 AM,440.0000
Sales Associate,Steven,Buchanan,Order ID,Shipped
Date,Sales,TOTAL,,75567.7500,10254,7/23/1996 12:00:00 AM,625.2000
Sales Associate,Steven,Buchanan,Order ID,Shipped
Date,Sales,TOTAL,,75567.7500,10269,8/9/1996 12:00:00 AM,676.0000
Sales Associate,Steven,Buchanan,Order ID,Shipped
Date,Sales,TOTAL,,75567.7500,10297,9/10/1996 12:00:00 AM,1420.0000
```

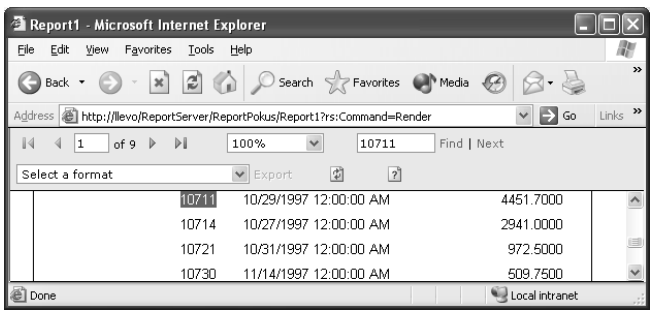
Pri exporte do súborových formátov (CSV, PDF, TIFF...) si podobne ako pri sťahovaní súborových dokumentov z webu môžeme vybrať, či chceme daný dokument len otvoriť, prípadne ho uložiť na disk klientského počítača pre následné použitie.



Obr. 3.41 Export reportu do formátu PDF (pre Acrobat Reader)



Obr. 3.42 Zobrazenie reportu vo formáte PDF pomocou prehliadača Acrobat Reader



Obr. 3.43 Vyhľadávanie v reporte

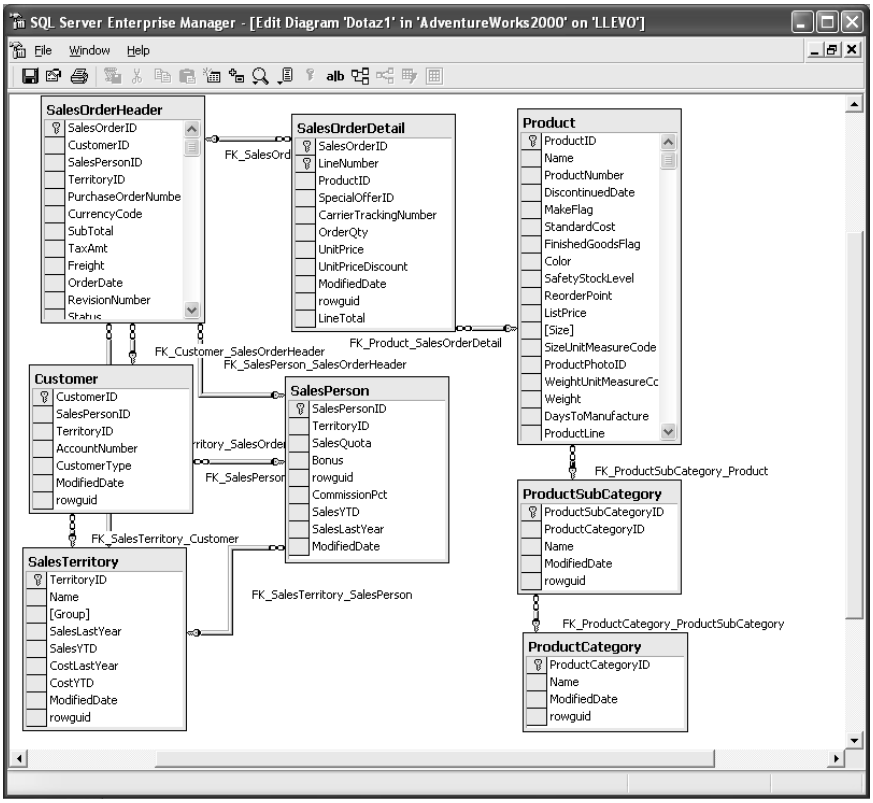
Report Designer – komplexný projekt

Účelom predchádzajúceho príkladu bolo v prvom rade zoznámenie sa nielen s vývojovým prostredím v režime Report Designer, ale hlavne kompletný postup vytvorenia jednoduchého reportu. Znovu opakujeme kompletný, nie komplexný. Zamerali sme sa na to, kde sa čo navrhuje, aké sú jednotlivé fázy návrhu a ich súslednosť. Jednoducho, cieľom bolo ukázať, ako sa to celé robí, pričom samotný námet reportu bol úplne triviálny. Príklad v tejto stati, teda konkrétne jeho námet bude už podstatne viac odrážať realitu.

SQL dopyt – základ budúceho reportu

Skôr než pristúpime k vytvoreniu reportu, je potrebné venovať sa jeho databázovej časti, t.j vytvoriť a odladiť SQL dopyt pre výber údajov, ktoré potrebujeme v reporte mať.

Príklad bude vytvorený nad cvičnou databázou AdventureWorks2000. Štruktúra tejto databázy a postup jej pripojenia pod správu databázového servera SQL Server 2000 sú popísané v prvej kapitoly. Databáza AdventureWorks2000 je pomerne rozsiahla. Na našom diagrame je len jej časť, konkrétne tabuľky, ktoré v dopyte využijeme



Obr. 3.44 Časť diagramu databázy AdventureWorks2000

Report bude obsahovať stĺpce `Region, Category, SubCategory, Sales, Cost` presnejšie v databázovej terminológii povedané prvé tri názvy sú aliasy stĺpcov pričom názvy stĺpcov z databázových tabuliek pre jednotlivé aliasy sú v tabuľke

Region	SalesTerritory.Name
Category	ProductCategory.Name
SubCategory	ProductSubCategory.Name

Štvrtý a piaty stĺpec sú výsledkom agregačných funkcií SUM

Sales	SUM(Product.ListPrice * SalesOrderDetail.OrderQty)
Cost	SUM(Product.StandardCost * SalesOrderDetail.OrderQty)

SQL dopyt pre výber údajov potom bude v tvare.

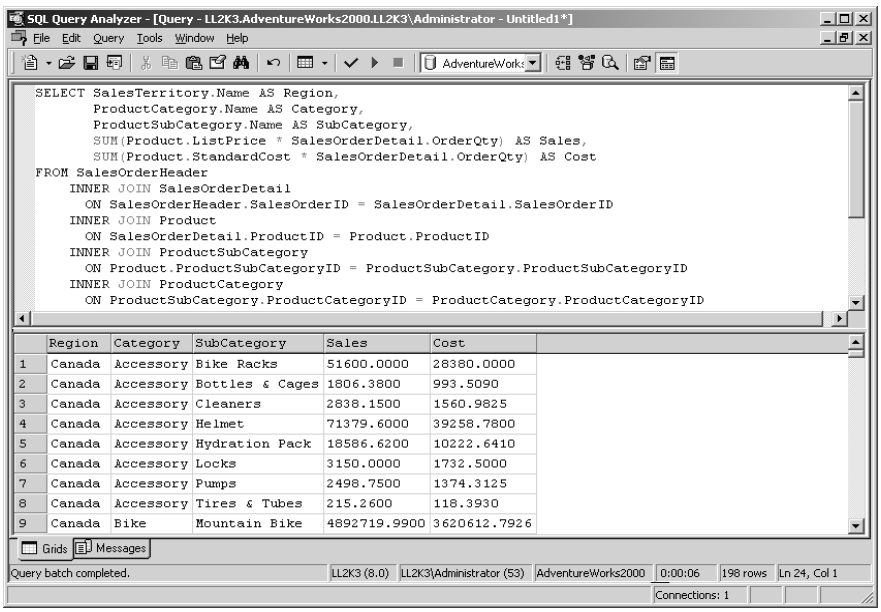
```
SELECT  SalesTerritory.Name AS Region,
        ProductCategory.Name AS Category,
        ProductSubCategory.Name AS SubCategory,
        SUM(Product.ListPrice * SalesOrderDetail.OrderQty) AS Sales,
        SUM(Product.StandardCost * SalesOrderDetail.OrderQty) AS Cost
FROM SalesOrderHeader
     INNER JOIN SalesOrderDetail
       ON SalesOrderHeader.SalesOrderID = SalesOrderDetail.SalesOrderID
     INNER JOIN Product
       ON SalesOrderDetail.ProductID = Product.ProductID
     INNER JOIN ProductSubCategory
       ON Product.ProductSubCategoryID = ProductSubCategory.ProductSubCategoryID
     INNER JOIN ProductCategory
       ON ProductSubCategory.ProductCategoryID = ProductCategory.ProductCategoryID
     INNER JOIN SalesPerson
       ON SalesPerson.SalesPersonID = SalesOrderHeader.SalesPersonID
     INNER JOIN SalesTerritory
       ON SalesPerson.TerritoryID = SalesTerritory.TerritoryID
     INNER JOIN Customer
       ON SalesOrderHeader.CustomerID = Customer.CustomerID
WHERE SalesTerritory.[Group] = 'North America'
  AND Customer.CustomerType = 'S'
GROUP BY SalesTerritory.Name, ProductCategory.Name, ProductSubCategory.Name
```

Výsledný výpis má 198 riadkov, preto ho uvádzame mierne skrátený, ale tak aby jeho informačná hodnota z hľadiska prehľadu o údajoch zostala zachovaná.

```
Region           Category       SubCategory          Sales                Cost
---------------  -------------  -------------------  -------------------  -------------
Canada           Accessory      Bike Racks           51600.0000           28380.0000
Canada           Accessory      Bottles & Cages      1806.3800            993.5090
Canada           Accessory      Cleaners             2838.1500            1560.9825
Canada           Accessory      Helmet               71379.6000           39258.7800
Canada           Accessory      Hydration Pack       18586.6200           10222.6410
Canada           Accessory      Locks                3150.0000            1732.5000
Canada           Accessory      Pumps                2498.7500            1374.3125
Canada           Accessory      Tires & Tubes        215.2600             118.3930
Canada           Bike           Mountain Bike        4892719.9900         3620612.7926
Canada           Bike           Road Bike            6682497.1700         4945047.9058
Canada           Bike           Touring Bike         2326077.1800         1721297.1132
Canada           Clothing       Bib-Short            37615.8200           20688.7010
Canada           Clothing       Cap                  8549.4900            4702.2195
Canada           Clothing       Gloves               42216.4700           23219.0585
Canada           Clothing       Jersey               151784.5300          83481.4915
Canada           Clothing       Shorts               61640.8100           33902.4455
Canada           Clothing       Socks                4617.6600            2539.7130
Canada           Clothing       Tights               39894.6800           21942.0740
Canada           Clothing       Vest                 58737.5000           32305.6250
Canada           Component      Bottom Bracket       7478.1900            5533.8606
Canada           Component      Brakes               11076.0000           8196.2400
...
Canada           Component      Touring Frame        329052.5300          243498.8722
Canada           Component      Wheel                93237.2500           68995.5650
Central          Accessory      Bike Racks           41040.0000           22572.0000
Central          Accessory      Bottles & Cages      1606.7800            883.7290
```

```
...
Northeast       Accessory       Bike Racks          19920.0000          10956.0000
Northeast       Accessory       Bottles & Cages     708.5800            389.7190
...
Northwest       Accessory       Bike Racks          21960.0000          12078.0000
Northwest       Accessory       Bottles & Cages     968.0600            532.4330
...
Southeast       Accessory       Bike Racks          17760.0000          9768.0000
Southeast       Accessory       Bottles & Cages     778.4400            428.1420
...
Southwest       Accessory       Bike Racks          57720.0000          31746.0000
Southwest       Accessory       Bottles & Cages     2190.6100           1204.8355
...

(198 row(s) affected)
```

Pripomíname, že SQL dopyt môžeme vyskúšať a odladiť v konzolovej aplikácie SQL Query Analyzer pred fázou návrhu, alebo pomocou nástroja Query Builder priamo v etape návrhu.



Obr. 3.45 Test SQL dopytu pomocou aplikácie Query Analyzer

Vytvorenie reportu pomocou sprievodcu

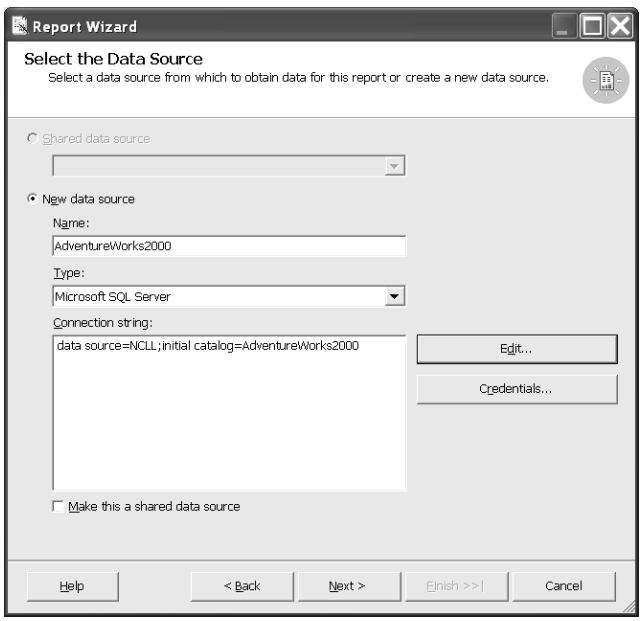
Podrobný postup založenia projektu typu Report Project je popísaný v predchádzajúcej stati. Teraz pôjdeme inou cestou, využijeme Report Project Wizard. Prvý krok je identický – vo vývojovom prostredí Visual Studio.NET 2003 aktivujeme položku menu pre vytvorenie nového projektu. V zložke typov projektov Business Intelligence Projects vyberieme projekt typu Report Project Wizard.

Obr. 3.46 Založenie nového projektu

Po zadaní názvu projektu, napríklad Sales Reports sa aktivuje sprievodca vytvorením reportu. Sprievodca nám bude nápomocný v nasledujúcich fázach návrhu
– výber dátového zdroja a nastavenie parametrov pre pripojenie sa k databáze
– vytvorenie, alebo grafický návrh SQL dopytu pre výber požadovanej množiny údajov
– výber typu reportu
– výber a návrh vzhľad reportu
– formátovanie textu, tabuliek a grafická úprava reportu

Obr. 3.47 Úvodný dialóg sprievodcu vytvorením reportu

Taktiež dialóg pre výber zdroja údajov je identický s predchádzajúcim príkladov. Rozdiel je len vo výbere zdroja údajov. Tento krát ním bude databáza AdventureWorks2000

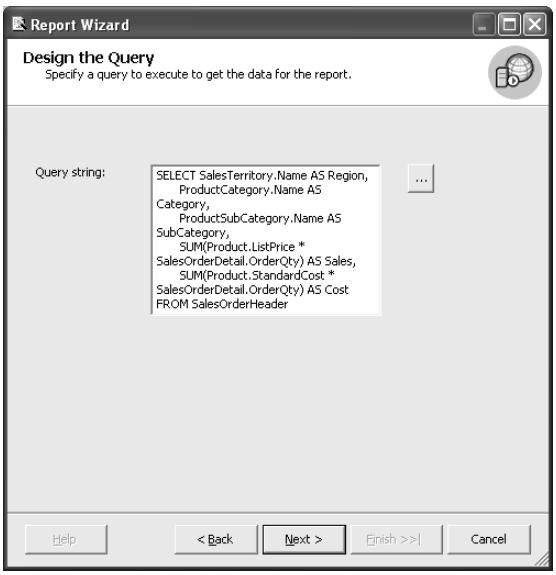


Obr. 3.48 Výber zdroja údajov

V okne dialógu vidíme sprievodcom vygenerovaný reťazec pre pripojenie sa k zdroju údajov.

```
provider=SQLOLEDB.1;
data source=localhost;
initial catalog=AdventureWorks2000;
integrated security=SSPI;
persist security info=False
```

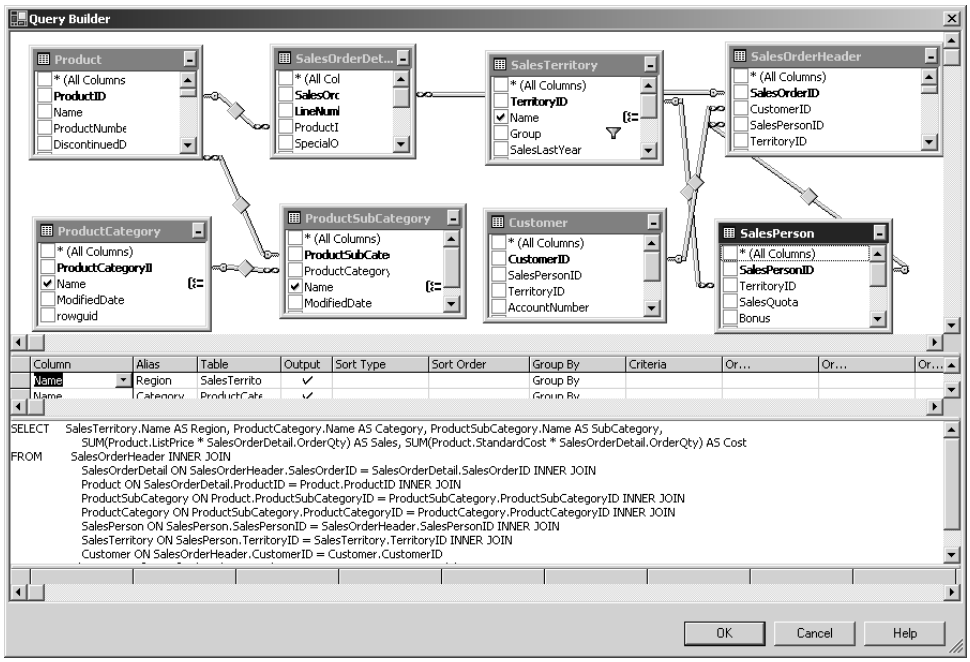
Po potvrdení údajov v dialógu „Select Data Source“ nasleduje pravdepodobne najzaujímavejšia, ale zároveň aj najnáročnejšia časť návrhu reportu – návrh SQL dopytu. Ak máme SQL dopyt vopred navrhnutý a vyskúšaný, čo je aj náš prípad, zadáme jeho textový reťazec do okna „Query String“ dialógu „Design the Query“



Obr. 3.49 Dialóg pre zadanie SQL dopytu

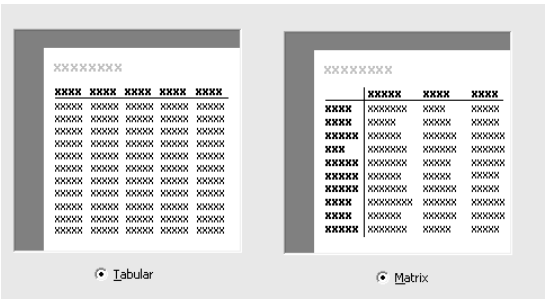
```
SELECT SalesTerritory.Name AS Region,
       ProductCategory.Name AS Category,
       ProductSubCategory.Name AS SubCategory,
       SUM(Product.ListPrice * SalesOrderDetail.OrderQty) AS Sales,
       SUM(Product.StandardCost * SalesOrderDetail.OrderQty) AS Cost
FROM SalesOrderHeader
    INNER JOIN SalesOrderDetail
      ON SalesOrderHeader.SalesOrderID = SalesOrderDetail.SalesOrderID
    INNER JOIN Product
      ON SalesOrderDetail.ProductID = Product.ProductID
    INNER JOIN ProductSubCategory
      ON Product.ProductSubCategoryID = ProductSubCategory.ProductSubCategoryID
    INNER JOIN ProductCategory
      ON ProductSubCategory.ProductCategoryID = ProductCategory.ProductCategoryID
    INNER JOIN SalesPerson
      ON SalesPerson.SalesPersonID = SalesOrderHeader.SalesPersonID
    INNER JOIN SalesTerritory
      ON SalesPerson.TerritoryID = SalesTerritory.TerritoryID
    INNER JOIN Customer
      ON SalesOrderHeader.CustomerID = Customer.CustomerID
WHERE SalesTerritory.[Group] = 'North America'
  AND Customer.CustomerType = 'S'
GROUP BY SalesTerritory.Name, ProductCategory.Name, ProductSubCategory.Name
```

SQl dopyt môžeme vytvoriť aj v etape návrhu pomocou nástroja Query Builder



Obr. 3.50 Vytvorenie SQL dopytu pomocou nástroja Query Builder

Nasleduje výber typu stránok reportu. K dispozícii je typ Tabular a Matrix. Pre náš projekt vyberieme možnosť **Tabular**.



Obr. 3.51 Ilustračná schéma výberu typu reportu Tabular /Matrix

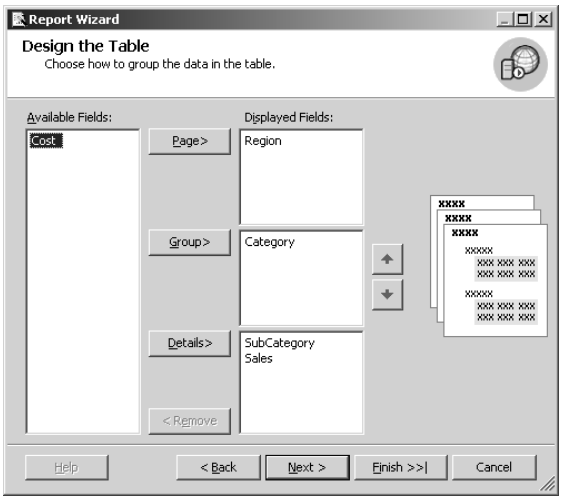
Podľa výberu v tomto kroku sa kroky sprievodcu vetvia, aj keď aj naďalej budú veľmi podobné. My sme si vybrali typ reportu Tabuľka, takže budeme pracovať s dialógom pre návrh štruktúry výslednej tabuľky (Design the Table). Princíp návrhu v tejto fáze je jednoduchý. Z poľa „Available Fields“ budeme presúvať potrebné položky do poľa „Displayed Fields“. Toto pole je rozdelené na tri časti

– Page
– Group
– Details

Pri tejto činnosti si musíme predstaviť, ako bude nami navrhovaný report vyzerať a podľa toho presúvať jednotlivé položky. V tejto činnosti, nám výrazne pomôže ikonka v pravej časti dialógu, kde sú vyznačené oblasti reportu v ktorých sa práve presúvané údaje objavia.

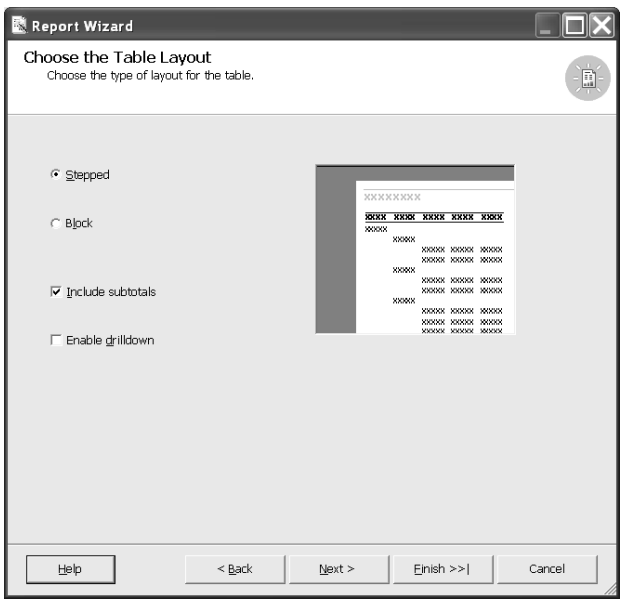
V našom príklade popresúvame názvy polí podľa tabuľky (vyplnený dialóg viď obrázok)

Page	Region
Group	Group
Details	Subcategory Sales



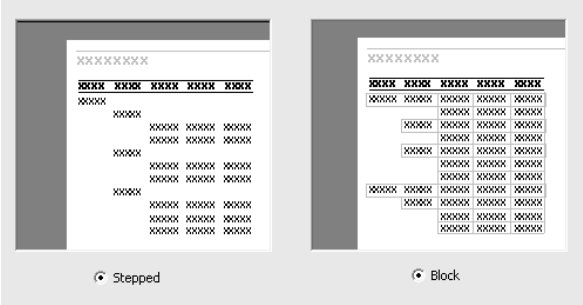
Obr. 3.52 Dialóg pre návrh štruktúry tabuľky

V dialógu „Choose the Table Layout“ vyberieme dizajn typu „Stepped“, pričom zaškrtneme políčko „Include subtotals“



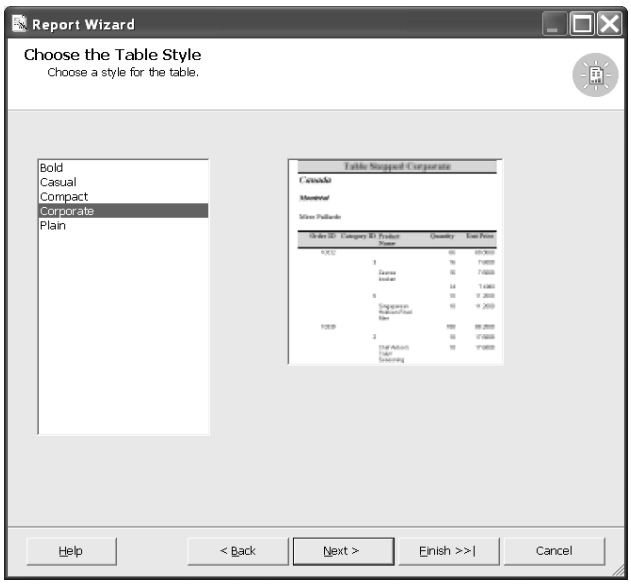
Obr. 3.53 Dialóg „Choose the Table Layout“

Budúci vzhľad reportu pre možnosť Stepped a zhustenú variantu Block je dostatočne zrejmý z ilustračného obrázku.



Obr. 3.54 Ilustračná schéma výberu typu schémy Stepped / Block

Návrh reportu v tejto fáze je viac otázkou štýlu, než obsahu a tak aj ďalší dialóg sa týka výberu štýlu tabuľky



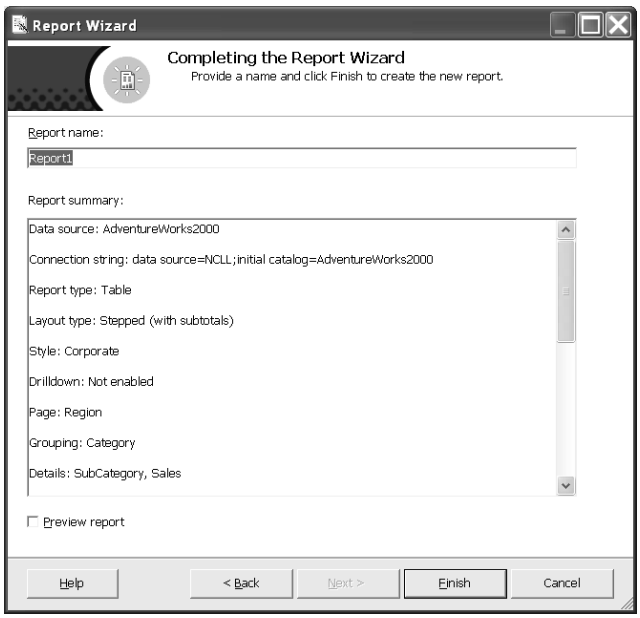
Obr. 3.55 Dialóg pre výber štýlu dokumentu

Na výber sú ponúkané možnosti
- Bold
- Casual
- Corporate
- Compact
- Plain

Jednotlivé možnosti opäť najlepšie predstaví ilustračný obrázok. My sme zvolili štýl Corporate

Obr. 3.56 Ilustračná schéma výberu štýlu dokumentu

Týmto krokom sme vlastne návrh reportu zavŕšili. V záverečnom dialógu si môžeme pozrieť sumárne informácie o tomto reporte. Report môžeme vhodne pomenovať.

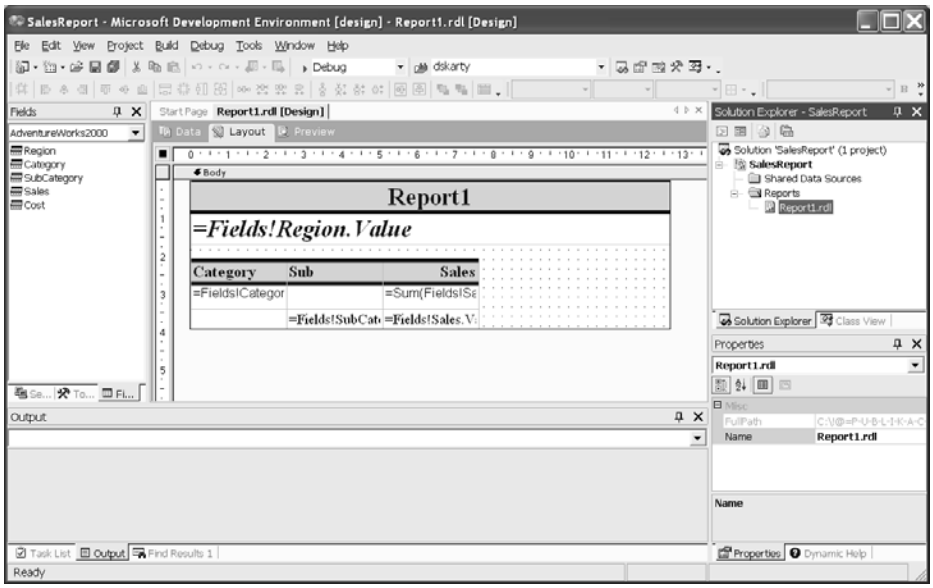


Obr. 3.57 Záverečný dialóg sprievodcu vytvorením reportu

Na záver návrhu sa poistíme. V menu vývojového prostredia File aktivujeme možnosť Save All.

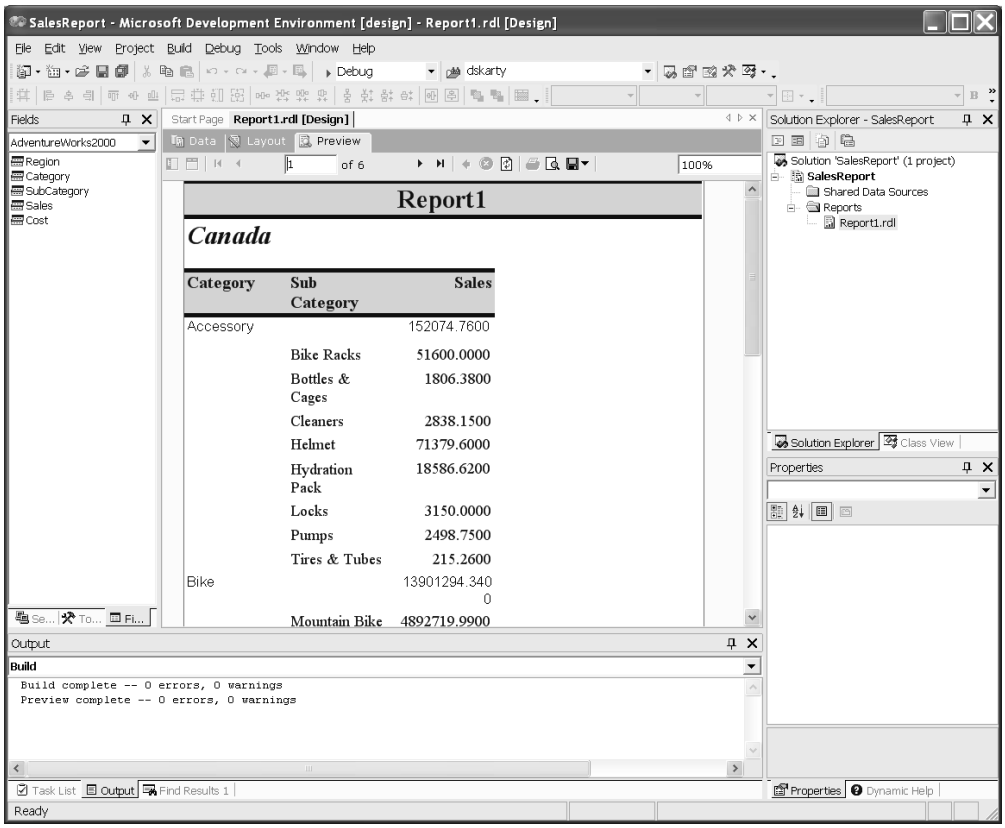
Finalizácia návrhu „obchodnej grafiky“ reportu

V predchádzajúcej fáze návrhu sme viac menej špecifikovali aké údaje v reporte chceme mať. Síce sme globálne definovali typ reportu, štýl tabuliek a podobne, o nejakom prispôsobení reportu našim požiadavkám zatiaľ nemôže byť ani reči. Skôr než pristúpime k finalizácii návrhu dizajnu reportu sa s ním trochu v tomto „surovom“ stave zoznámime. V strednom zvislom okne pracovnej obrazovky Report Designera sú ikony pre prepínanie jeho módov. Môžeme prepínať medzi návrhovým zobrazením, to je stav, v ktorom sa vývojové prostredie nachádza po ukončení práce sprievodc vytvorením reportu



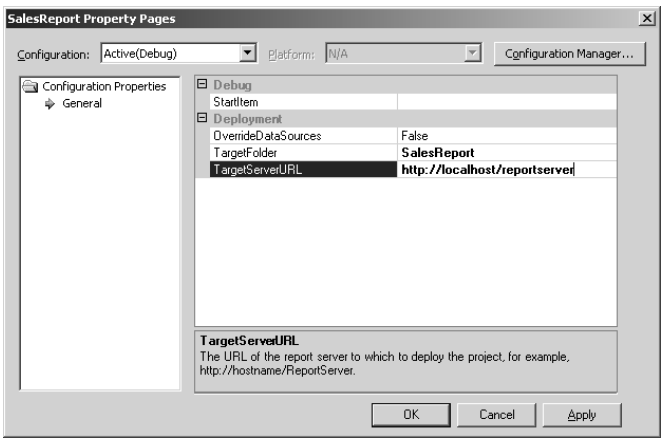
Obr. 3.58 Report Designer v režime Layout

a zobrazením Preview, v ktorom si môžeme prehliadnuť finálnu podobu report tak, ako bude v budúcnosti zobrazený u jeho „konzumentov“. Stredné okno vývojového prostredia sa v mode prewiew prepne do režimu prehliadača webového obsah



Obr. 3.59 Report Designer v režime Preview

Ak všetko funguje tak ako má (pripomíname, že zatiaľ nás zajíma len obsah, konečný dizajn reportu budeme ešte dotvárať), môžeme report zostaviť a odoslať na reportovací server (menu Build, Deploy Soltion), kde bude k dispozícii pre klientov. Predtým však ešte musíme nastaviť URL adresu servera (kontextové menu na názov projektu v okne Solution Explorer).



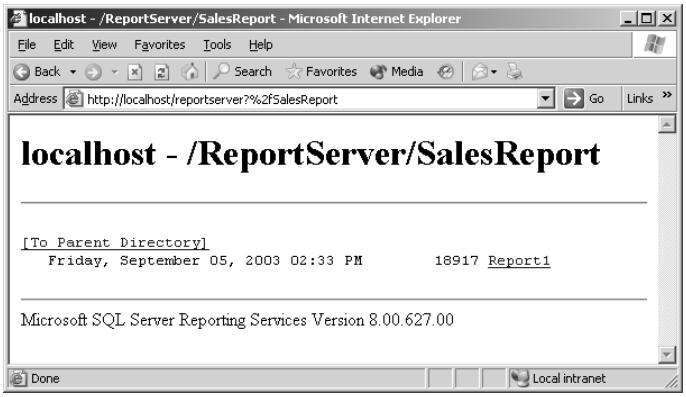
Obr. 3.60 Nastavenie URL adresy

V okne Output si všimnime protokol o zostavení, kde je aj URL adresa, na ktorej je report prístupný.

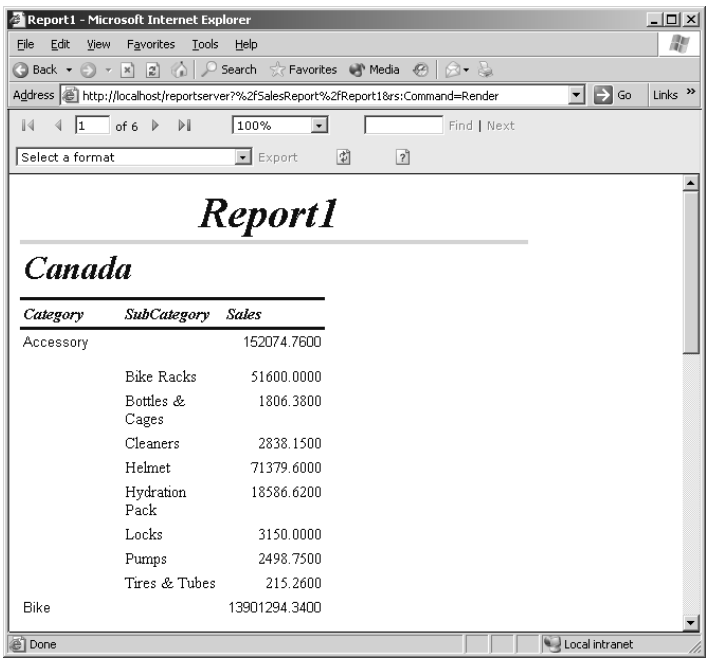
```
------ Build started: Project: SalesReport, Configuration: Debug ------

Build complete -- 0 errors, 0 warnings

------ Deploy started: Project: SalesReport, Configuration: Debug ------

Deploying to http://localhost/reportserver?%2fSalesReport
Deploying report 'Report1'
```

```
Deploy complete -- 0 errors, 0 warnings

---------------------- Done ----------------------

    Build: 1 succeeded, 0 failed, 0 skipped
    Deploy: 1 succeeded, 0 failed, 0 skipped
```

Túto URL adresu môžeme zadať do prehliadača Internet Explorer a prezrieť si momentálnu podobu reportu, tak ako ju reportovací server poskytuje klientom.

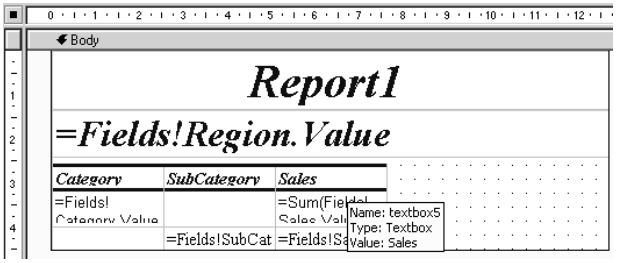


Obr. 3.61 Test reportu pomocou webového prehliadača – zoznam reportov



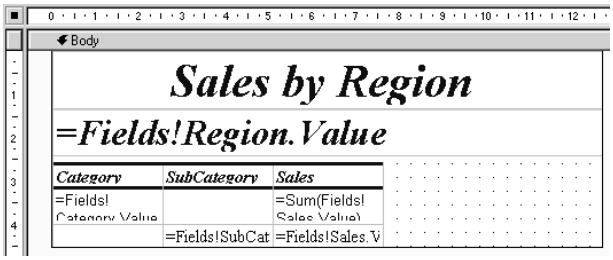
Obr. 3.62 Test reportu pomocou webového prehliadača - zobrazenie reportu

Report je pomerne zrozumiteľný, prehľadný a má dosť dobrú úpravu, no pravdepodobnosť, že úplne vyhovie našim požiadavkám je pomerne malá. Sprievodca si na základe nami zadaných údajov urobil svoju robotu, no myšlienky čítať nevie. A tak si report prispôsobíme sami. Vychádzame z počiatočného stavu, ktorý vidíte na obrázku a dúfam že aj na obrazovkách svojich počítačov.



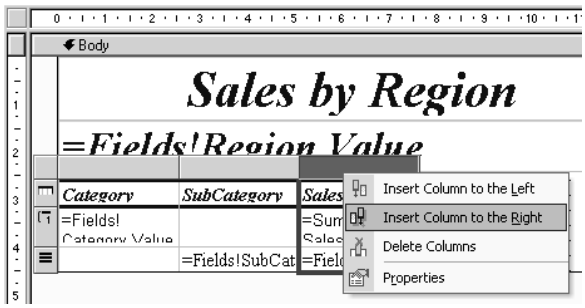
Obr. 3.63 Report vytvorený sprievodcom

Najskôr zmeníme názov reportu Názov Report1 sme či už úmyselne, alebo nedopatrením ponechali v príslušnom návrhovom dialógu ako názov reportu. Nakoľko databáza AdventureWorks2000 je anglická a tak aj názvy stĺpcov pri automatickom návrhu pomocou sprievodcu sú anglické, slovenský alebo český názov reportu by situáciu rozhodne nevylepšil. Preto náš report nazveme „Sales by region“. Ak by sme predsa len chceli, môžeme počeštiť alebo poslovenčiť celý report, napriek tomu, že je postavený na anglickej databáze.



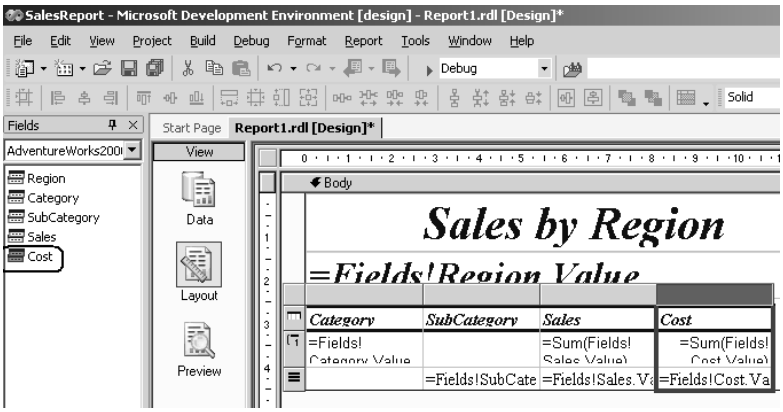
Obr. 3.64 Zmena názvu reportu

Nasledovať bude pridanie ďalšieho stĺpca do reportu. Postup je jednoduchý. Označíme stĺpec, s ktorým bude budúci stĺpec susediť a pomocou kontextového menu určíme, či novo vytvorený stĺpec bude situovaný vľavo, alebo vpravo od označeného stĺpca.



Obr. 3.65 Vytvorenie nového stĺpca

Po potvrdení jednej zo spomínaných možností sa vytvorí prázdny stĺpec. Následne napíšeme do záhlavia názov stĺpca („Cost“) a pole Cost presunieme z ľavého okna Fields do druhého a tretieho riadku nového stĺpca v návrhového zobrazenia.



Obr. 3.66 Návrh nového stĺpca

Overenie správnosti pridania nového stĺpca je triviálne. Stačí sa prepnúť do okna Preview.

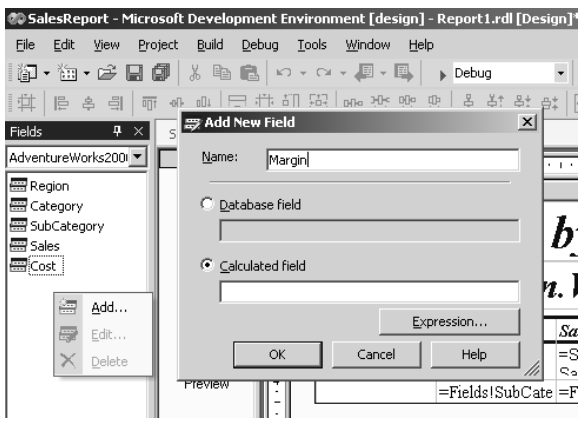
Sales by Region

Canada

Category	SubCategory	Sales	Cost
Accessory		152074.7600	83641.1180
	Bike Racks	51600.0000	28380.0000
	Bottles & Cages	1806.3800	993.5090
	Cleaners	2838.1500	1560.9825
	Helmet	71379.6000	39258.7800
	Hydration Pack	18586.6200	10222.6410
	Locks	3150.0000	1732.5000
	Pumps	2498.7500	1374.3125
	Tires & Tubes	215.2600	118.3930

Obr. 3.67 Návrh nového stĺpca

Ak si pozrieme v Report Designeri zoznam polí (okno toolboxu „Fields“), zistíme, že všetky dostupné polia, teda stĺpce databázových tabuliek, ktoré sme vybrali v etape návrhu SQL dopytu sme v reporte už použili. No napriek tejto skutočnosti sa možnosti návrhu reportu nekončia, skôr začínajú. Do reportu môžeme napríklad umiestniť vypočítané polia. V okne Fields aktivujeme položku kontextového menu „Add“ a v dialógu „Add New field“ zadáme názov nového poľa napríklad „Margin“ (v našom cvičnom príklade pre jednoduchosť už u tej angličtiny zostaneme) a označíme prepínač „Calculated field“



Obr. 3.68 Návrh nového stĺpca

Pre skonštruovanie výrazu slúži dialóg aktivovaný tlačidlom „Expression...“

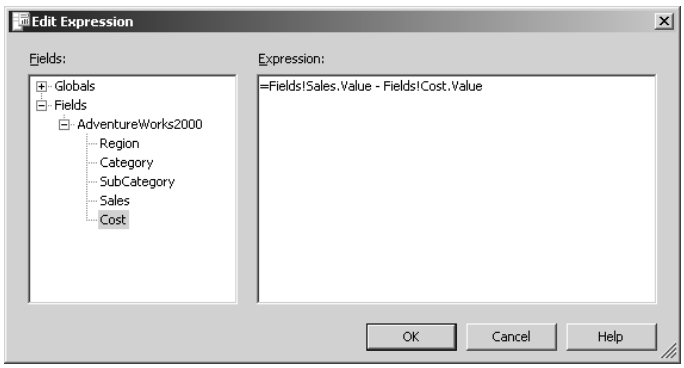
Vzorec pre naše vypočítané pole je

MARGIN = SALES - COST

jeho prepis do tvaru zrozumiteľného pre Report Designer, teda do jazyka RDL bude

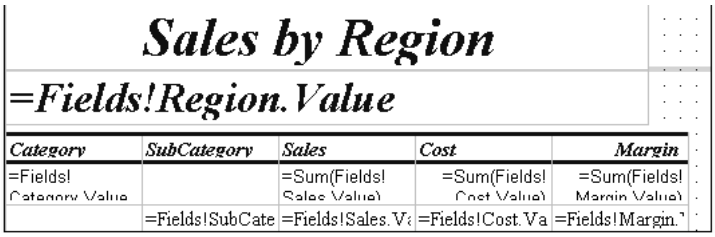
```
=Fields!Sales.Value - Fields!Cost.Value
```

Tento jednoduchý výraz vy sme mohli pokojne vpísať do editačného okienka. Vyskúšajme si však možnosti ktoré poskytuje nástroj „Edit Expression“. V poli výrazu je zatiaľ len =. Z okna polí presunieme pole SALES, pripíšeme znamienko mínus, presunieme pole COST a návrh výrazu je hotový. Pomáha nám to hlavne pri dodržiavaní syntaxe.



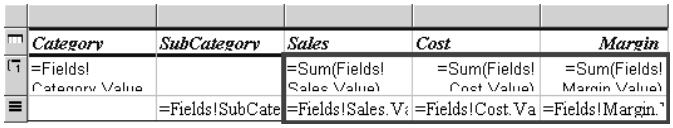
Obr. 3.69 Návrh výrazu pre vypočítaný stĺpec

V okne Fields sa objaví ikonka nami navrhnutého vypočítaného stĺpca MARGIN. Pri jeho pridaní do návrhu reportu postupujeme rovnako ako sme postupovali pri pridaní nového stĺpca COST



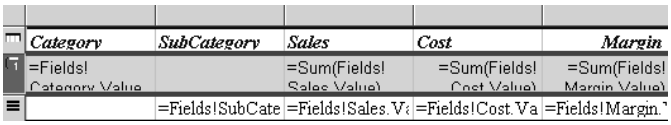
Obr. 3.70 Pridanie vypočítaného stĺpca

Pozrime sa podrobnejšie na výpis údajov v tabuľke reportu, prakticky všetky údajové polia (na obrázku označené čiernym obdĺžnikom) obsahujú údaje v menových jednotkách. V reporte sú zatiaľ zobrazené pomerne správne ako desatinné číslo s presnosťou na štyri desatinné miesta. No toto zobrazenie sa samozrejme dá vylepšiť. Stačí ak tieto polia označíme a v okne Properties (pravé dolné okno pracovnej obrazovky vývojového prostredia) nastavíme vlastnosť formát na hodnotu C (z anglického currency).



Obr. 3.71 Formátovanie vybraných polí

Možno by sme chceli zvýrazniť niektoré riadky, napríklad tie v ktorých sú agregované sumy. Môžeme pre ne nastaviť napríklad šedé pozadie.



Obr. 3.72 Zvýraznenie niektorých riadkov reportu zmenou farby pozadia

Výsledok našej doterajšej snahy si môžeme prezrieť v okne Preview. Pre američana je výsledok vynikajúci, pre našinca, ktorý neobchoduje s USA je formát meny trochu neobvyklý, ale tak to proste v USA chodí...

Sales by Region

Canada

Category	SubCategory	Sales	Cost	Margin
Accessory		$152,074.76	$83,641.12	$68,433.64
	Bike Racks	$51,600.00	$28,380.00	$23,220.00
	Bottles & Cages	$1,806.38	$993.51	$812.87
	Cleaners	$2,838.15	$1,560.98	$1,277.17
	Helmet	$71,379.60	$39,258.78	$32,120.82
	Hydration Pack	$18,586.62	$10,222.64	$8,363.98
	Locks	$3,150.00	$1,732.50	$1,417.50
	Pumps	$2,498.75	$1,374.31	$1,124.44
	Tires & Tubes	$215.26	$118.39	$96.87
Bike		$13,901,294.34	$10,286,957.81	$3,614,336.53
	Mountain Bike	$4,892,719.99	$3,620,612.79	$1,272,107.20
	Road Bike	$6,682,497.17	$4,945,047.91	$1,737,449.26

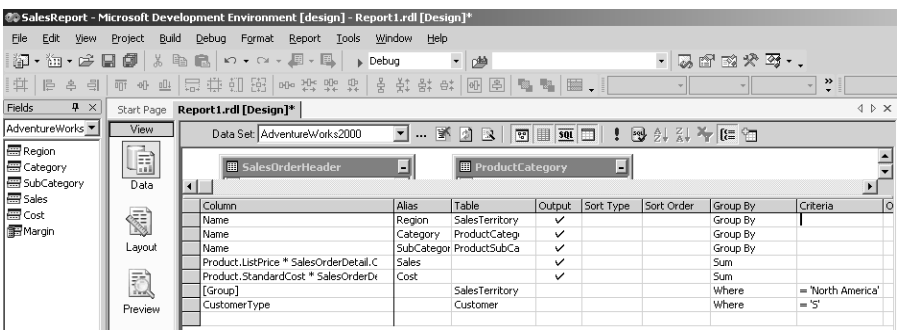
Obr. 3.73 Preview reportu po zmenách formátovania

Interaktívny report

Pokiaľ by sme potrebovali náš účel len pre účelu prehliadania, alebo tlače, je report presne v takom stave ako potrebujeme. No klienti napríklad webových aplikácií, ktorí sú zvyknutí pristupovať k rôznym výpisom a prehľadom považujú za samozrejmé, aby rôzne reporty a prehľady boli dynamické a hlavne interaktívne, to znamená aby sme mali možnosť napríklad nechať si zobraziť len tú podmnožinu údajov, ktorú potrebujeme. Ak chceme vytvoriť interaktívny report, musíme sa vrátiť k jeho samotnému jadru na ktorom je každý report vybudovaný – k SQL dopytu. Ak chceme rôzne špecifikovať množinu dopytom vrátených údajov, potrebujeme zaviesť do SQL dopytu parametre.

Dosiaľ sme v návrhovom prostredí pracovali v módoch Layout a Preview. Pre parametrizáciu SQL dopytu sa prepneme do režimu Data a v strednej časti pracovnej obrazovku si všimnime tabuľku a v nej stĺpec Criteria. V tabuľke sú kritériá nastavené pre riadky

Column	Criteria
[Group]	= 'North America'
CustomerType	= 'S'



Obr. 3.74 Pracovná obrazovka vývojového prostredia v režime „Data“

Čo znamenajú parametre v stĺpci „Criteria“ by sme mohli siahodlho vysvetľovať, no stačí si pripomenúť SQL dopyt, konkrétne hrubo časť vytlačenú hrubým písmom.

```
SELECT SalesTerritory.Name AS Region,
       ProductCategory.Name AS Category,
       ProductSubCategory.Name AS SubCategory,
       SUM(Product.ListPrice * SalesOrderDetail.OrderQty) AS Sales,
       SUM(Product.StandardCost * SalesOrderDetail.OrderQty) AS Cost
FROM SalesOrderHeader
    INNER JOIN SalesOrderDetail
      ON SalesOrderHeader.SalesOrderID = SalesOrderDetail.SalesOrderID
    INNER JOIN Product
      ON SalesOrderDetail.ProductID = Product.ProductID
    INNER JOIN ProductSubCategory
      ON Product.ProductSubCategoryID = ProductSubCategory.ProductSubCategoryID
    INNER JOIN ProductCategory
      ON ProductSubCategory.ProductCategoryID = ProductCategory.ProductCategoryID
    INNER JOIN SalesPerson
      ON SalesPerson.SalesPersonID = SalesOrderHeader.SalesPersonID
    INNER JOIN SalesTerritory
      ON SalesPerson.TerritoryID = SalesTerritory.TerritoryID
    INNER JOIN Customer
      ON SalesOrderHeader.CustomerID = Customer.CustomerID
WHERE SalesTerritory.[Group] = 'North America'
  AND Customer.CustomerType = 'S'
GROUP BY SalesTerritory.Name, ProductCategory.Name, ProductSubCategory.Name
```

V riadku [Group] nahradíme reťazec **= 'North America'** reťazcom **=@Teritory**.

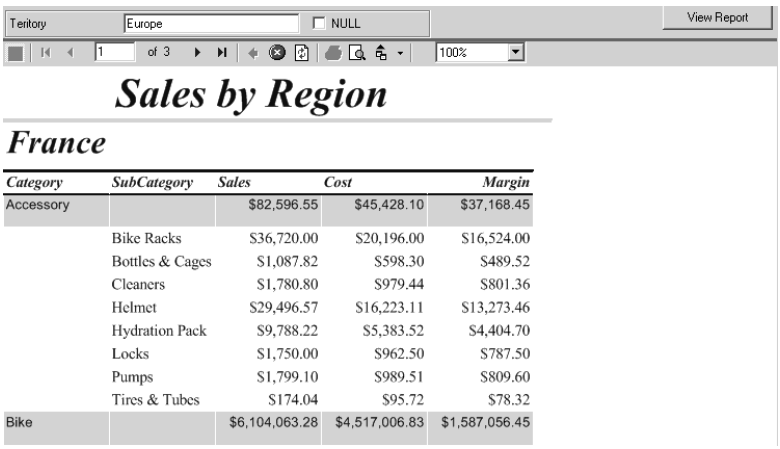
Column	Alias	Table	Output	Sort Type	Sort Order	Group By	Criteria
Name	Region	SalesTerritory	✓			Group By	
Name	Category	ProductCateg	✓			Group By	
Name	SubCategor	ProductSubCa	✓			Group By	
Product.ListPrice * SalesOrderDetail.C	Sales		✓			Sum	
Product.StandardCost * SalesOrderDe	Cost		✓			Sum	
[Group]		SalesTerritory				Where	=@Teritory
CustomerType		Customer				Where	= 'S'

Obr. 3.75 Tabuľka pre návrh dopytu

Stačí tento jednoduchý návrhový úkon a už môžeme v reporte zadávať parametre. Ak aktivujeme režim preview, zobrazí sa nám najskôr prázdny report, len s lištou pre zadanie parametra. Ak sa na lištu podívame pozornejšie, zistíme, prečo je report prázdny. Vedľa textboxu pre zadanie parametra je zašktrávacie políčko NULL, ktoré je zaškrtnuté. Predmetná časť podmienky potom znie

```
WHERE SalesTerritory.[Group] = NULL
```

a tejto podmienke nevyhovuje ani jeden záznam. Ak zrušíme zaškrtnutie políčka NULL, môžeme do textboxu zadávať parametre. Zadajme napríklad parameter Europe.



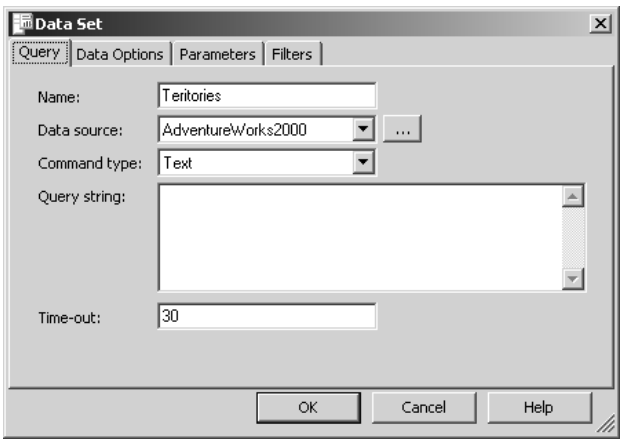
Category	SubCategory	Sales	Cost	Margin
Accessory		$82,596.55	$45,428.10	$37,168.45
	Bike Racks	$36,720.00	$20,196.00	$16,524.00
	Bottles & Cages	$1,087.82	$598.30	$489.52
	Cleaners	$1,780.80	$979.44	$801.36
	Helmet	$29,496.57	$16,223.11	$13,273.46
	Hydration Pack	$9,788.22	$5,383.52	$4,404.70
	Locks	$1,750.00	$962.50	$787.50
	Pumps	$1,799.10	$989.51	$809.60
	Tires & Tubes	$174.04	$95.72	$78.32
Bike		$6,104,063.28	$4,517,006.83	$1,587,056.45

Obr. 3.76 Preview reportu s parametrom

Predchádzajúca ukážka bola jednoduchá a pomerne efektná. Až na to zadávanie názvu teritória. Priznám sa, skúšal som rôzne teritória (na angličtine sme sa predsa názvy svetadielov učili), napríklad „Asia“, „Africa“, no úspech som zožal málokedy. Oveľa lepšie by bolo zadávať teritória pomocou Comboboxu. No od niekiaľ ich musíme vziať. Napojme teda report na tabuľku SalesTeritory a zistime názvy teritórií z jej stĺpca Group. ľahko sa povie, ale o niečo ťažšie realizuje.

Pre prácu s viacerými a to aj nezávislými tabuľkami využijeme bohaté možnosti, ktoré poskytuje DataSet. DataSet e základná a zároveň najkomplexnejšia trieda v rozhraní ADO.Net pre prácu s údajmi. Z hľadiska uloženia údajov a manipulácie s nimi je to v podstate relačná databáza v pamäti. Údajmi sa plní buď z databázy pomocou objektu DataAdapter, prípadne pracuje s údajmi vo formáte XML. Dôležité je pochopiť, že DataSet je ako taký úplne odpojený od dátového zdroja, alebo inými slovami povedané, on sám sa nikdy nepripojí priamo do databázy. DataSet vlastne ani nevie z akého druhu dátového zdroja pochádzajú, či z XML, Oracle, alebo z SQL Servera. Takáto univerzálnosť je vynikajúca z hľadiska škálovateľnosti a prípadnej migrácie aplikácie. DataSet môže laicky povedané zapúzdrovať niekoľko nehomogénnych databázových tabuliek, presnejšie DataSet pozostáva z viacerých objektov typu DataTable. Tieto objekty majú podobné vlastnosti ako tabuľky v relačnom databázovom serveri. (riadky, stĺpce, integritné obmedzenia, relačné vzťahy medzi tabuľkami cez cudzie kľúče...).

Report môže dokonca obsahovať viacero datasetov, pričom každý môže byť napojený na rôzne dátové zdroje. Nový dataset vytvoríme tak, že v hornej lište pracovnej obrazovky v mode Data aktivujeme možnosť „New Data Set“. Dataset nazveme napríklad „teritories“ a bude napojený taktiež na databázu AdventureWorks2000.

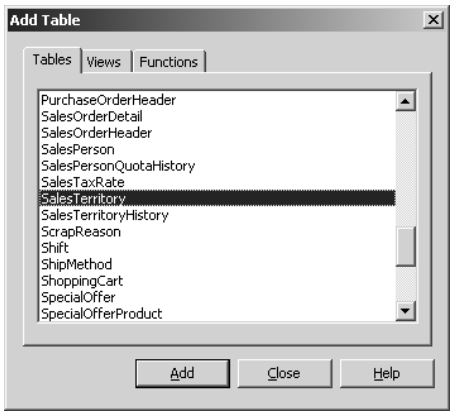


Obr. 3.77 Vytvorenie nového datasetu

Pole v toolbare pre zadanie SQL dopytu ponecháme zatiaľ prázdne, tabuľky do novovytvoreného datasetu budeme pridávať inak. Combobox Data Set ponecháme nastavený na názve nového datasetu „Teritories“ a aktivujeme tlačidlo „Add Table“ (posledné tlačidlo toolbaru vpravo)

Obr. 3.78 Pridanie novej tabuľky do datasetu tlačidlom „Add Table“

Do datasetu pridáme tabuľku SalesTeritoru



Obr. 3.79 Pridanie tabuľky SalesTeritory do datasetu

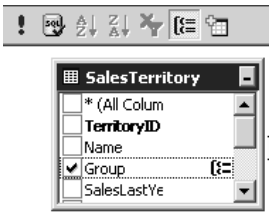
Pre zaujímavosť si vypíšeme názvy stĺpcov tejto tabuľky a ich dátové typy

TerritoryID	tinyint
Name	Name
Group	nvarchar (50)
SalesLastYear	money
SalesYTD	money
CostLastYear	money
CostYTD	money
ModifiedDate	datetime

Môžeme si nechať pomocou aplikácie Query Analyzer vypísať obsah tejto tabuľky SQL dopytom

```
SELECT TerritoryID, Name, Group, SalesLastYear, SalesYTD FROM SalesTerritory

TerritoryID  Name            Group          SalesLastYear  SalesYTD
-----------  --------------- -------------- -------------- ----------------
1            Northwest       North America  3298694.4938   4358526.0429
2            Northeast       North America  3607148.9371   4825355.2670
3            Central         North America  3205014.0767   3985928.6549
4            Southwest       North America  5366575.7098   6643143.7089
5            Southeast       North America  3925071.4318   4201294.9543
6            Canada          North America  5693988.8600   6169098.6679
7            France          Europe         2396539.7601   2177055.6488
8            Germany         Europe         1307949.7917   2160347.3087
9            Australia       Pacific        2278548.9776   2486869.8048
10           United Kingdom  Europe         1635823.3967   2568244.0549

(10 row(s) affected)
```

Zaujímať nás bude stĺpec Group, ktorý označíme a aplikujeme naň funkciu GROUP BY.
Čiže zoskupíme údaje pre Severnú Ameriku, Európu a Pacifickú oblasť



Obr. 3.80 Označenie stĺpca Group a aplikovanie GROUP BY na tento stĺpec

Týmito návrhovými úkonmi vznikne SQL dopyt

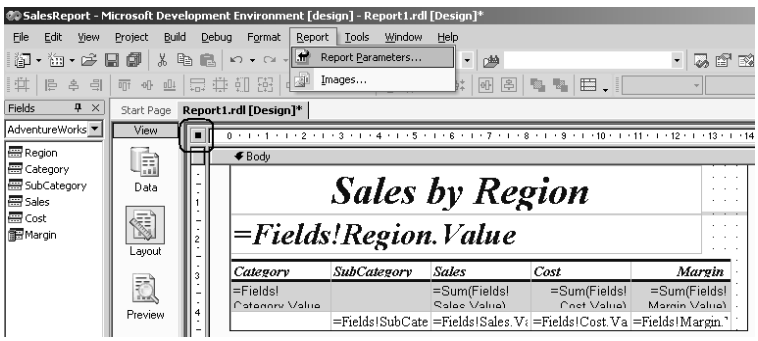
```
SELECT [Group] FROM SalesTerritory GROUP BY [Group]
```

Výsledok, ktorý takto vznikne, teda výpis teritórií by priniesol nové informácie leda tak päťkárovi zo zemepisu,

```
Group
--------------------
Europe
North America
Pacific
```

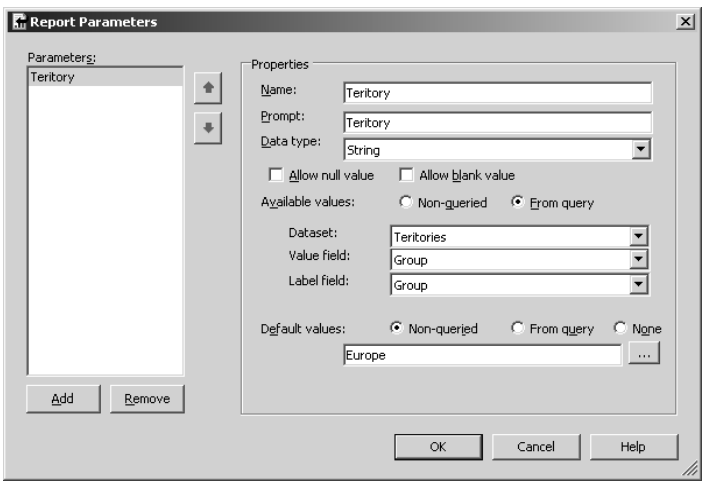
no počítač zemepis neštudoval a tak sa názvy regiónov od niekiaľ dozvedieť musel

Prepnime sa do módu Layout a nastavme report do východiskovej polohy. Túto akciu vykonáme jednoducho tak, že klikneme na malý štvorček (označený na obrázku) v ľavom hornom rohu návrhového formulára na priesečníku riadkových a stĺpcových pravítok. Následne aktivujeme v hlavnom menu položku „Report“ a v nej „Report Parameters“



Obr. 3.81 Aktivovanie položky menu „Parametre reportu“

V dialógu „Report Parameters“ zmeníme niektoré nastavenia. Zrušíme zaškrtnutie políčka „Allow null value“, a prepínač „Available values“ nastavíme na hodnotu „From Query“. Tým sa mierne zmení vzhľad dialógu. Pomocou comboboxu „dataset“ nastavíme tento parameter na dataset „Teritories“ a prepínač „Default values“ prepneme na hodnotu „Non-queried“



Obr. 3.82 Aktivovanie položky menu „Parametre reportu“

teraz po uložení návrhu (tlačidlo toolbaru Save All) sa môžeme opäť prepnúť do režimu „Preview“ a prepínanie regiónov si vyskúšať.

Obr. 3.83 Prepínanie parametrov

V návrhu „interaktivity“ reportu môžeme pokračovať. Určite ste si už v okne Preview všimli, že výpis obsahuje všetky detaily a z toho dôvodu je pomerne dlhý a nie úplne prehľadný. Prínosom by bolo začať výpisom agregovaných hodnôt zosumarizovaných pre bicykle, príslušenstvo, oblečenie a súčiastky. Získame tým komplexnú informáciu. Ale ak má klient záujem zistiť detailné údaje, mali by sme mu to umožniť. Túto časť návrhu vysvetlíme netradične odzadu. Najprv ukážeme výsledok návrhu a potom postupnosť krokov, ako sme toho dosiahli.

Sales by Region

France

Category	SubCategory	Sales	Cost	Margin
Accessory		$82,596.55	$45,428.10	$37,168.45
Bike		$6,104,063.28	$4,517,006.83	$1,587,056.45
Clothing		$216,355.25	$118,995.39	$97,359.86
Component		$1,464,389.29	$1,083,648.07	$380,741.22

Obr. 3.84 Report zo zakrytým výpisom detailných údajov

V sumárnom reporte si všimnime znamienka (+) vedľa nadpisu stĺpca „Category“. Ak klient na toto políčko klikne, môže si prehliadať podrobnejšie údaje a teda sa „zavŕta“ (Drill – down) do podrobností.

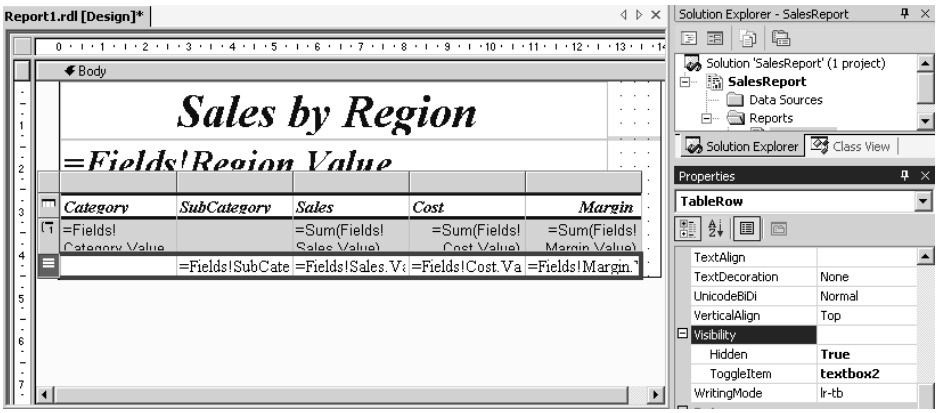
Sales by Region

France

Category	SubCategory	Sales	Cost	Margin
Accessory		$82,596.55	$45,428.10	$37,168.45
	Bike Racks	$36,720.00	$20,196.00	$16,524.00
	Bottles & Cages	$1,087.82	$598.30	$489.52
	Cleaners	$1,780.80	$979.44	$801.36
	Helmet	$29,496.57	$16,223.11	$13,273.46
	Hydration Pack	$9,788.22	$5,383.52	$4,404.70

Obr. 3.85 Report z rozvinutým výpisom detailov

Ak predpokladáme, že návrh takejto funkcionality reportu je zložitý, môžem vás ukľudniť, uvidíte, aké je to v Report Designeri jednoduché. Najskôr označíme v návrhovom zobrazení dolný riadok (kliknutím na ikonu ležiacu na jeho začiatku) a v okne „Properties“ sa budeme venovať nastaveniu parametra „Visibility“. Ešte predtým si musíme zistiť jednu maličkosť – názov textboxu, ku ktorému pridáme ovládací prvok pre rozvinutie detailov. Klikneme preto na pole nadpisu stĺpca „Category“ a zistíme a zapamätáme si jeho ID. V našom prípade to bolo textbox2. Teraz v okne „Properties“ rozvinieme parameter „Visibility“ a parameter Hidden nastavíme na hodnotu true (detaily budú implicitne skryté) a parameter ToggleItem nastavíme na ID spomínaného textboxu, v našom prípade to bol textbox2. Týmto jednoduchým krokom sme vlastne dosiahli celú funkcionalitu zobrazovania, prípadne potláčania zobrazovania detailných údajov.



Obr. 3.86 Prepínanie parametrov

Týmto krokom po odoslaní reportu pod správu reportovacieho servera považujeme návrh jednoduchého interaktívneho reportu za ukončený a môžeme ho „vydať napospas“ klientom. Kiežby to bolo také jednoduché. Je to analogický stav, ako keby konštruktéri oceľovej konštrukcie, napríklad mostu ho odovzdali do používania a.... odišli. Spočiatku by bolo všetko v najlepšom poriadku, no konštrukciu by pomaly alebo isto zožierať hrdza. Na konštrukciu pôsobia rôzne vplyvy, ktoré spôsobujú jej postupnú deštrukciu. Podobne je to aj s reportom. Po zavedení na server by sa chvíľu klienti tešili z kvalitne navrhnutých a dobre fungujúcich reportov, no po čase niekto zmení názov niektorého stĺpca v niektorej tabuľke, ktorú používame pri reportovaní a...

Obr. 3.87 Chyba

...naplno sa prejaví jav zvaný entropia. Nebudeme rozoberať fyziku ale zjednodušene sa entropia dá vysvetliť, že veci ponechané na seba spejú od desiatim k piatim.

V predchádzajúcej kapitole o architektúre reportovacích služieb sme zdôrazňovali význam troch fáz životného cyklu reportu

- Návrh
- Správa
- Doručenie

Preto ak sa chcete dozvedieť viac o správe reportov a v neposlednom rade aj o správe reportu, ktorý sme práve navrhli, prečítajte si ďalšiu kapitolou

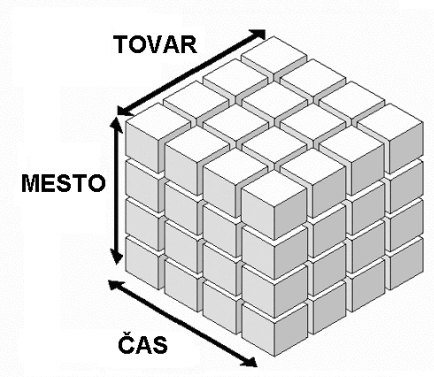
Reporty z analytických databáz

Dosiaľ sme sa zaoberali návrhom reportov nad relačnými databázami. Ako už bolo spomenuté SQL Server 2000 sa dodáva spolu s analytickými službami a OLAP serverom. Podrobne sa touto problematikou zaoberať nebudeme, je témou iného zborníka od firmy Microsoft a tak uvedieme len stručné základy OLAP

Údaje v klasickej relačnej databáze sú organizované v dvojrozmerných relačných tabuľkách. Každý riadok takejto tabuľky sa vzťahuje k nejakému predmetu, udalosti, alebo k ich časti. Na rozdiel od dvojrozmerného relačného modelu, multidimenzionálny databázový model si môžeme najjednoduchšie predstaviť ako priestorovú kocku. Kocka je vlastne ekvivalentom tabuľky v relačnej databáze. Každá kocka má niekoľko dimenzií. Na rozdiel od geometrickej kocky môže mať multidimenzionálny databázový model aj viac dimenzií ako tri. Príkladom typického trojdimenzionálneho modelu môže byť kocka s dimenziami:

- čas
- mesto
- tovar

Kocka sa v konvenčnom programovacom jazyku správa ako pole. Logicky je priestor pre celú kocku vopred rozvrhnutý. Údaje sa nachádzajú v prienikoch jednotlivých dimenzií. Údaje v kocke sú po jej vytvorení a pravidelných aktualizáciách okamžite k dispozícii pre rôzne analýzy.



Obr. 3.88 olap KOCKA

Môžeme analyzovať údaje len za určité časové obdobie, napríklad za účelom vyhodnotenia výsledkov reklamnej kampane, alebo sledovanosti webovej stránky za určité obdobie a podobne. Iným príkladom môžu byť údaje z určitého regiónu, ku ktorým má prístup regionálny riaditeľ, pre potreby jeho rozhodovania.

V podnikoch analytické služby pracujú s údajmi v produkčných databázach, alebo ešte lepšie s údajmi v dátových skladoch. Tieto činnosti vykonáva analytický server na výkonných podnikových serveroch. Server je však len určitá služba na pozadí, ktorá pripraví a spracuje údaje a vykoná požadované analýzy. K výsledkom analýz pristupujú kompetentné osoby, teda manažéri, analytici, finančnící a podobne pomocou rôznych klientských aplikácií.

Pre prácu s údajmi v analytických databázach sa používa jazyk MDX. Skratka MDX je akronym od slovného spojenia **M**ulti**d**imensional E**x**presions (multidimenzionálne výrazy). Jazyk MDX pre multidimenzionálne databázy je určitým ekvivalentom jazyka SQL v relačných databázach. Jeho primárnym cieľom je navigácia v multidimenzionálnych údajoch. Pomocou MDX dopytu vyšpecifikujeme určitú podmnožinu údajov z multidimenzionálnej štruktúry (OLAP kocky) a vypíšeme ju do dvojrozmernej tabuľky, ktorá obsahuje množinu buniek, preto tejto štruktúre hovoríme Cellset. A dvojrozmerné tabuľky – na to sa predsa hodia reportovacie služby.

Analógia jazyka MDX a SQL oblasťou použitia nekončí. Aj štruktúra príkazu pre výber údajov SELECT je podobná. Potešiteľné je, že jazyk MDX je v porovnaní s jazykom SQL podstatne jednoduchší. Syntax príkazu SELECT (podotýkame že úplná) pozostáva z klauzúl FROM a WHERE

```
SELECT  [<specifikace_osi>
        [, <specifikace_osi>...]]
  FROM  [<specifikace_kocky>]
[WHERE  [<specifikace_rezu>]]
```

Prvé pokusy s príkazmi jazyka MDX môžeme robiť pomocou aplikácie MDX Sample Application, ktorá sa nainštaluje spolu s analytickými službami SQL Serveru 2000 a spúšťa sa cez menu operačného systému v zložke aplikácií SQL servera. Po úspešnom prihlásení máme k dispozícii pomerne komfortný nástroj pre prácu s multidimenzionálnymi štruktúrami, jednoduchú konzolovú aplikáciu pomocou ktorej môžeme zadávať príkazy a prezerať si výsledky.

Pre našu ukážku reportu nad OLAP databázou vyberieme cvičnú kocku SALES analytickej databázy FoodMart2000 (nainštaluje sa spolu s analytickými službami)

Cvičná kocka Sales má 12 dimenzií.

Názov dimenzie	**Úrovne**	**Popis**
Customers	Country, State or Province, City, Name	Geografiká dimenzia, ktorá umožňuje skúmať, odkiaľ pochádzajú zákaznící
Education Level	Education Level	Úroveň vzdelania zákazníkov „Graduate Degree“ alebo „High School Degree.“
Gender	Gender	Pohlavie zákazníka: „M“ (zákazník) alebo „F“ (zákazníčka)
Marital Status	Marital Status	Rodinný stav zákazníka: „S“ (single – slobodný) alebo „M“ (ženatý, vydatá)
Product	Product Family Product Department Product Category Product Subcategory Brand Name Product Name	Produkty s ktorými spoločnosť FoodMart obchoduje.
Promotion Media	Media Type	Typ média používaného na propagáciu, napríklad denná tlač, rozhlas, televízia...
Promotions	Promotion Name	Ktorá reklama bola dôvodom k nákupu.
Store	Store Country Store State Store City Store Name	Geografiká dimenzia hierarchie obchodného reťazca.
Store Size in SQFT	Store Square Feet	Oblasť pôsobnosti predajne (v štvorcových stopách).
Store Type	Store Type	Typ predajne, napríklad „Deluxe Supermarket“ alebo „Small Grocery.“
Time	Years, Quarters, Months	Čas kedy sa obchod uskutočnil.
Yearly Income	Yearly Income	Ročný príjem zákazníka.

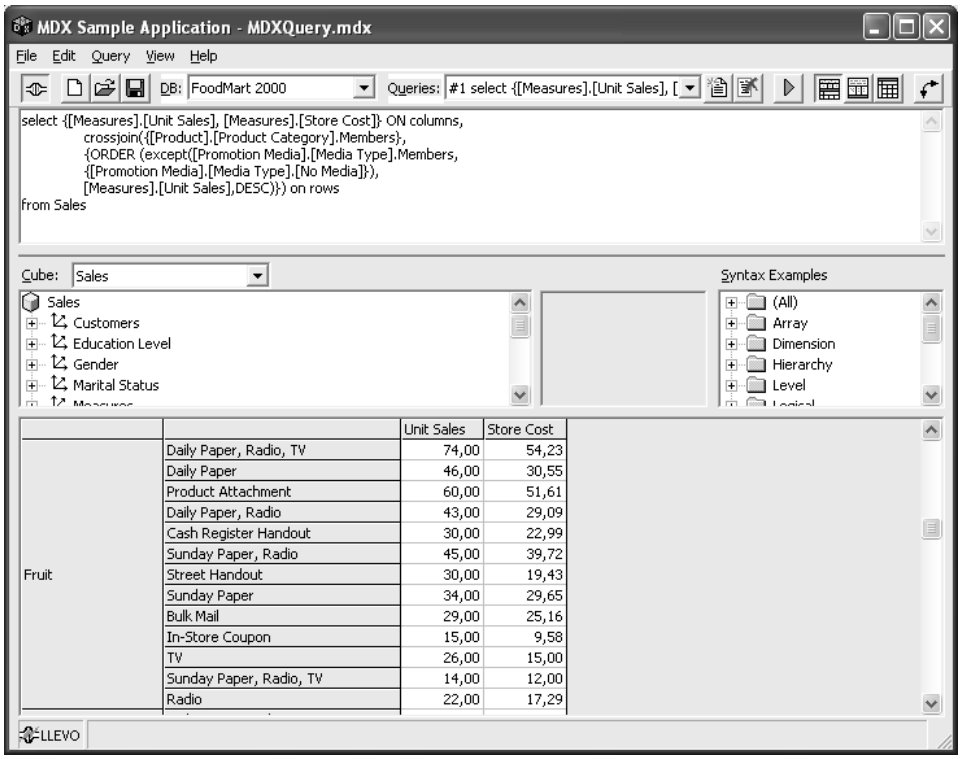
a obsahuje 6 mierok

Názov mierky	**Popis**
Unit Sales	Počet predaných kusov.
Store Cost	Hodnota predaného tovaru.
Store Sales	Hodnota obchodných transakcií.
Sales Count	Počet obchodných transakcií.
Store Sales Net	Hodnota transakcií za zníženú cenu
Sales Average	Predané množstvo/počet obchodov. (Vypočítaná mierka.)

Vytvoríme MDX dopyt napríklad

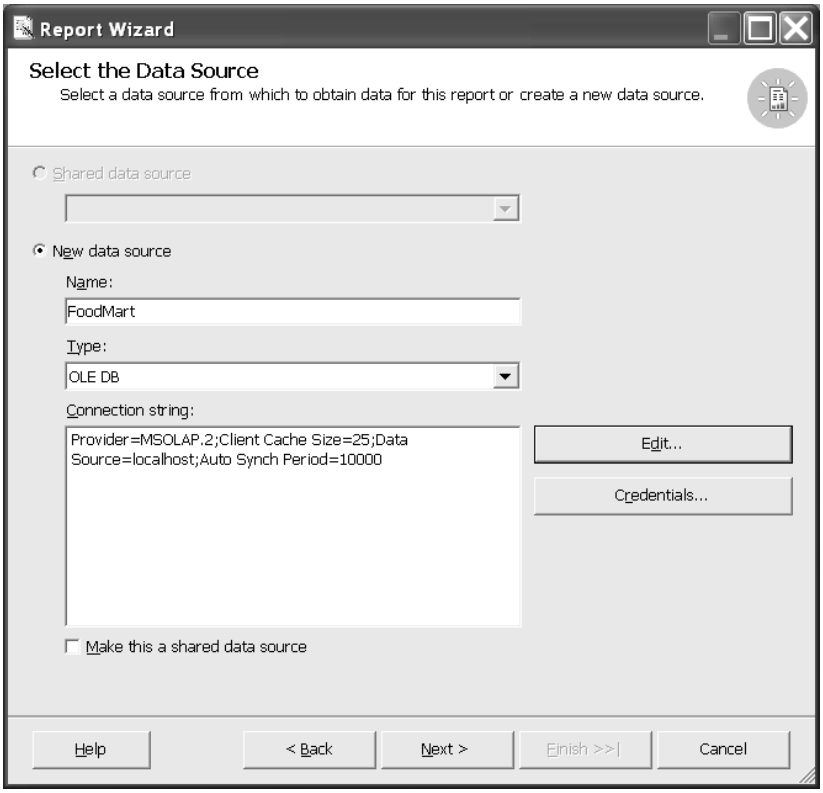
```
select {[Measures].[Unit Sales], [Measures].[Store Cost]} ON columns,
       crossjoin({[Product].[Product Category].Members},
       {ORDER (except([Promotion Media].[Media Type].Members,
       {[Promotion Media].[Media Type].[No Media]}),
       [Measures].[Unit Sales],DESC)}) on rows
from Sales
```

a hneď si ho môžeme vyskúšať pomocou MDX Sample Application



Obr. 3.89 – MDX Sample Application

Vyskúšaný a odladený MDX dopyt môžeme použiť pre návrh reportu. Ukážeme príklad reportu nad OLAP kockou, ktorá je pod správou MS OLAP Servera.

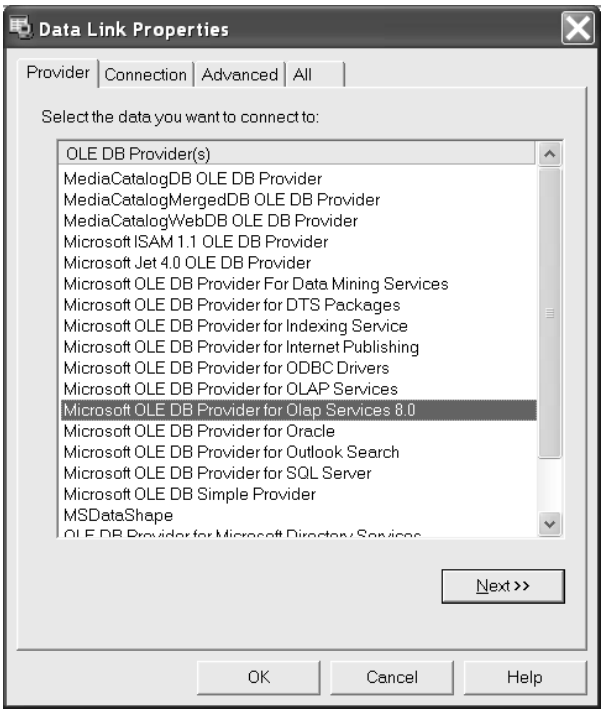


Obr. 3.90 – Analytická databáza ako zdroj údajov.

Pre pripojenie sa k OLAP serveru použijeme Microsoft OLE DB Provider for Olap Services. Do okna pre pripojovací reťazec zadáme

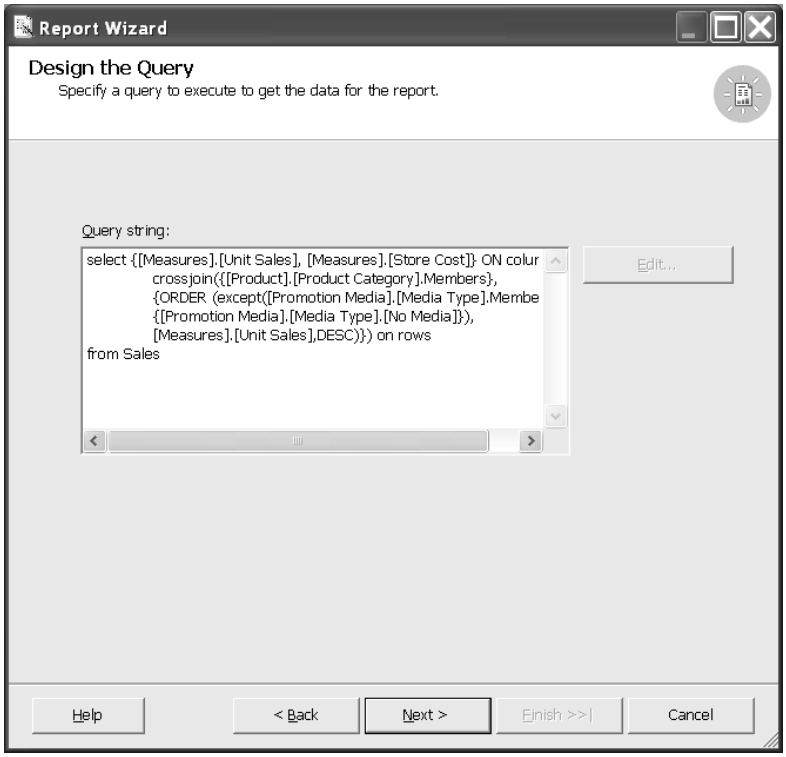
```
Provider=MSOLAP.2;Client Cache Size=25;Data Source=localhost;Auto Synch Period=10000
```

Prípadne príslušný provider (Microsoft OLE DB Provider for Olap Services) nastavíme pomocou dialógu Data Link Properties.



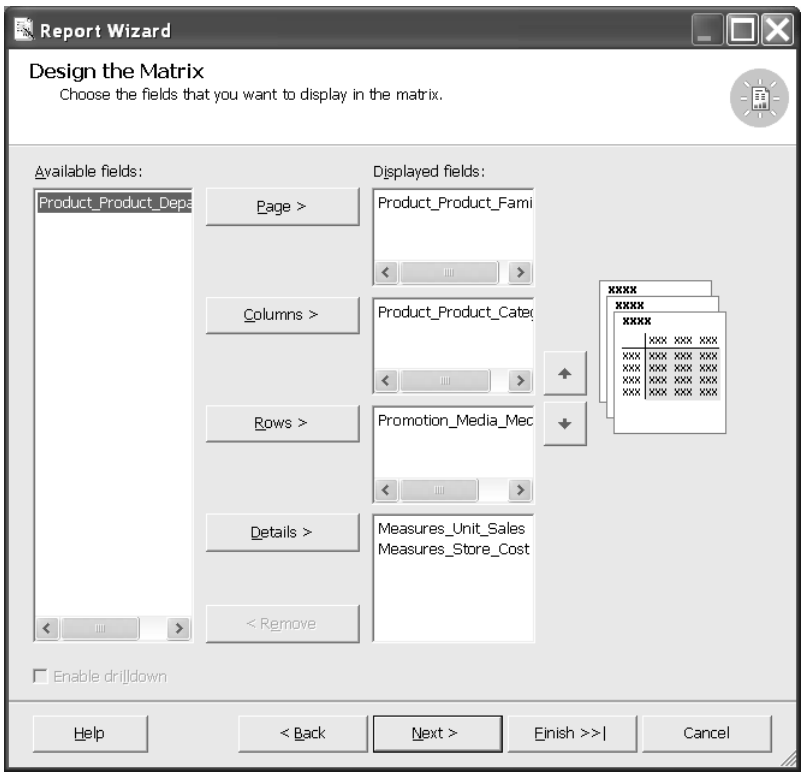
Obr. 3.91 – Výber providera pre pripojenie sa k analytickej databáze

V dialógu „Design the Query“ zadáme MDX dopyt. Všimnime si, že tlačidlo Edit je zablokované, grafické interaktívne nástroje na aké sme zvyknutí pri tvorbe SQL dopytov pre tvorbu MDX dopytov k dispozícii v tejto verzii nie sú.



Obr. 3.92 – Zadanie MDX dopytu

V tejto fáze návrhu už musíme mať predstavu o vzhľade budúceho reportu. Podľa toho volíme jeho základný look buď ako Table, alebo ako Matrix. Túto predstavu zhmotníme v kostru reportu pomocou dialógu pre návrh tabuľky, alebo matice. Vyberieme príslušné polia, ktoré chceme mať v riadkoch stĺpcoch a v detailných bunkách.

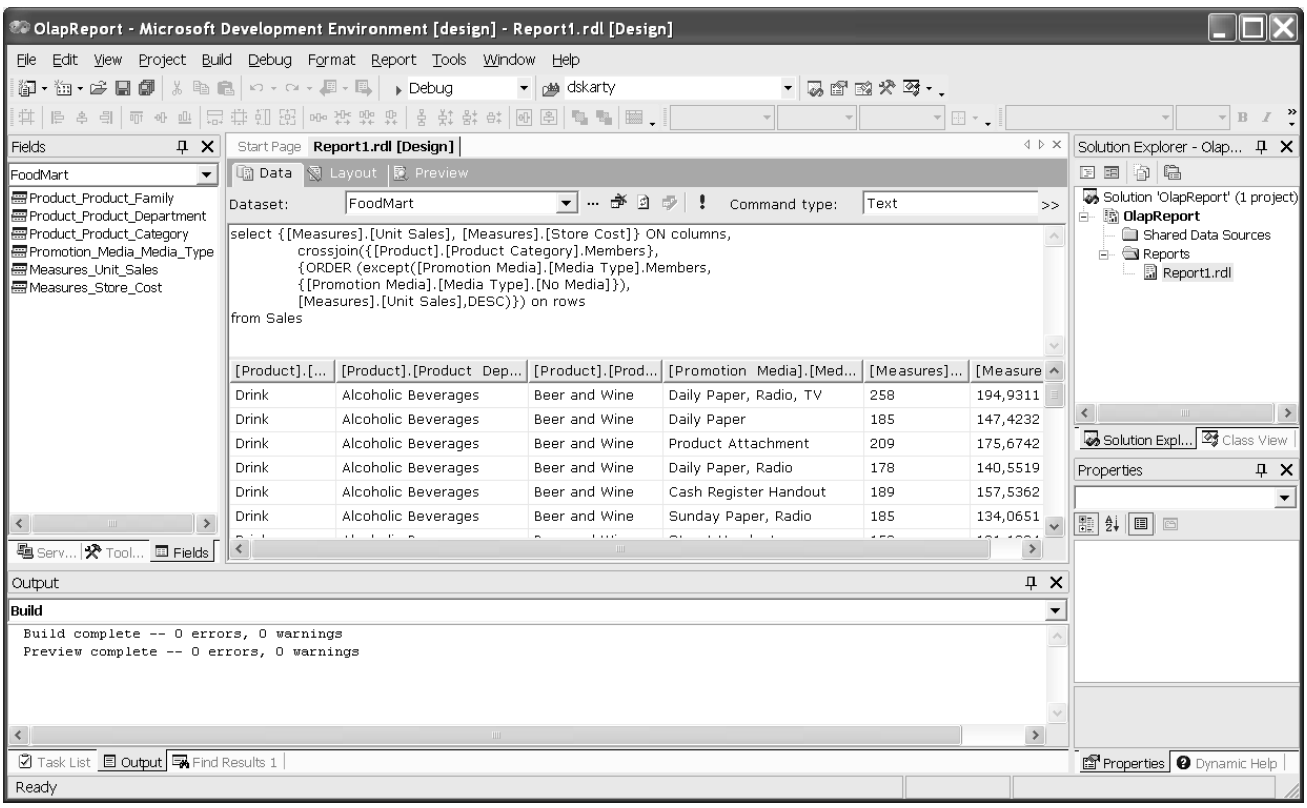


Obr. 3.93 – Návrh tabuľky alebo matice

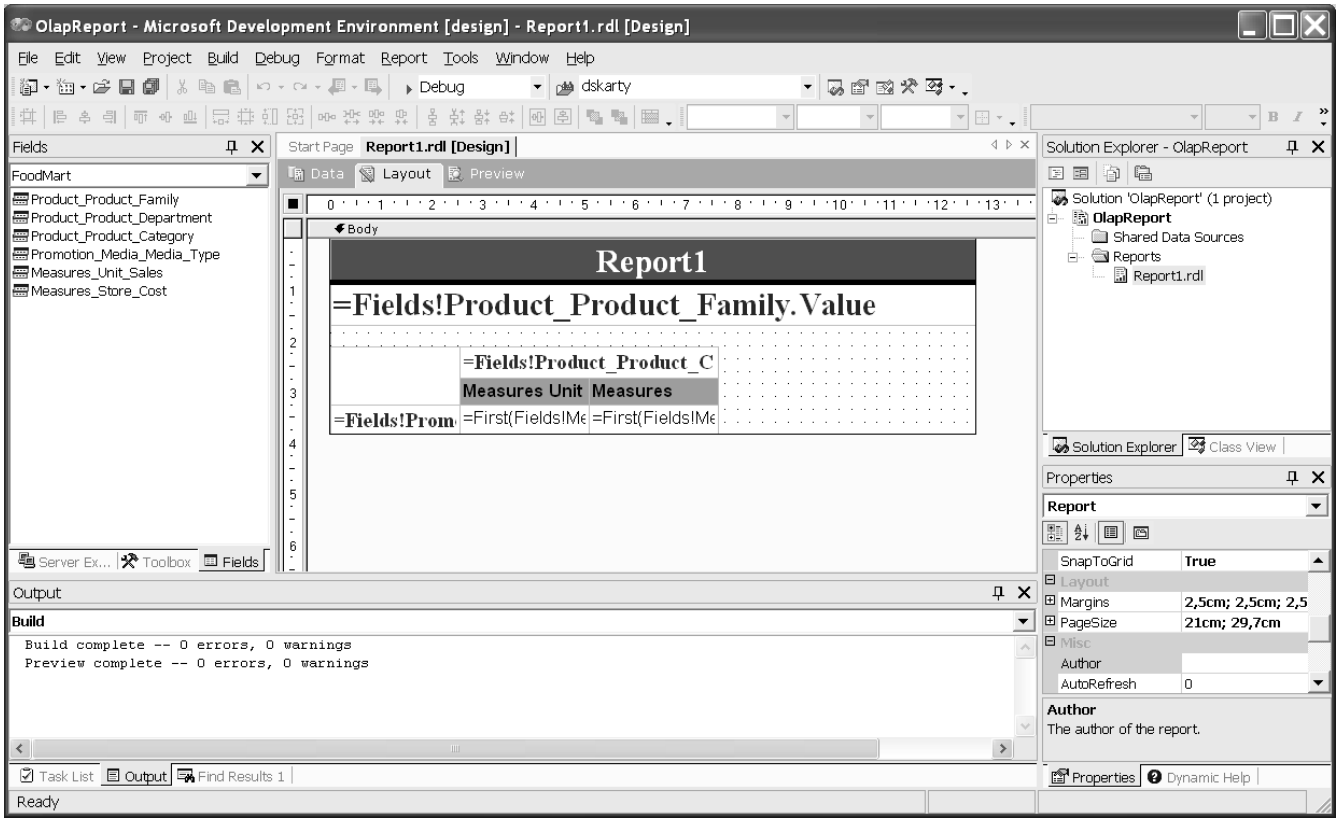
V poslednom dialógu sprievodcu vytvorením reportu sa zobrazia informácie o poliach, stĺpcoch a riadkoch a zobrazí sa aj Query, teda MDX dopyt.

```
Data source: FoodMart
Connection string: Provider=MSOLAP.2;Client Cache Size=25;Data Source=localhost;Auto
Synch Period=10000
Report type: Matrix
Style: Bold
Page: Product_Product_Family
Column: Product_Product_Category
Row: Promotion_Media_Media_Type
Details: Measures_Unit_Sales, Measures_Store_Cost

Query:  select {[Measures].[Unit Sales], [Measures].[Store Cost]} ON columns,
       crossjoin({[Product].[Product Category].Members},
       {ORDER (except([Promotion Media].[Media Type].Members,
       {[Promotion Media].[Media Type].[No Media]}),
       [Measures].[Unit Sales],DESC)}) on rows
```

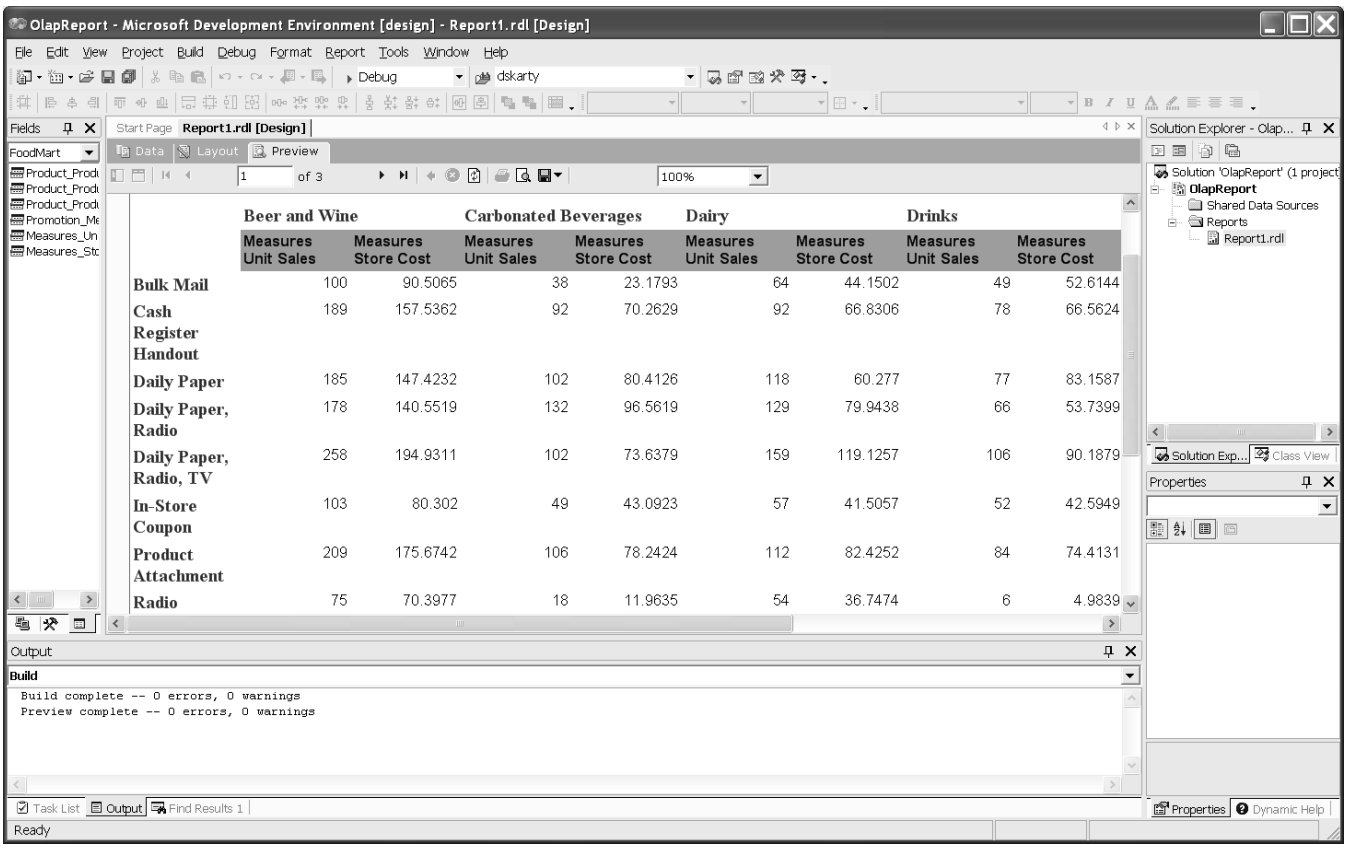
Ak by sme to mali zovšeobecniť, návrh reportu nad analytickými údajmi je úplne rovnaký ako návrh reportu nad relačnými údajmi. Rozdiel je len vo výbere údajov pomocou dopytu. Zatiaľ čo u relačných databáz sa pre definovanie výberu údajov používa SQL dopyt, pre definovanie výberu údajov z multidimenzionálnych databáz sa používa MDX dopyt.



Obr. 3.94 – OLAP report – zložka Data



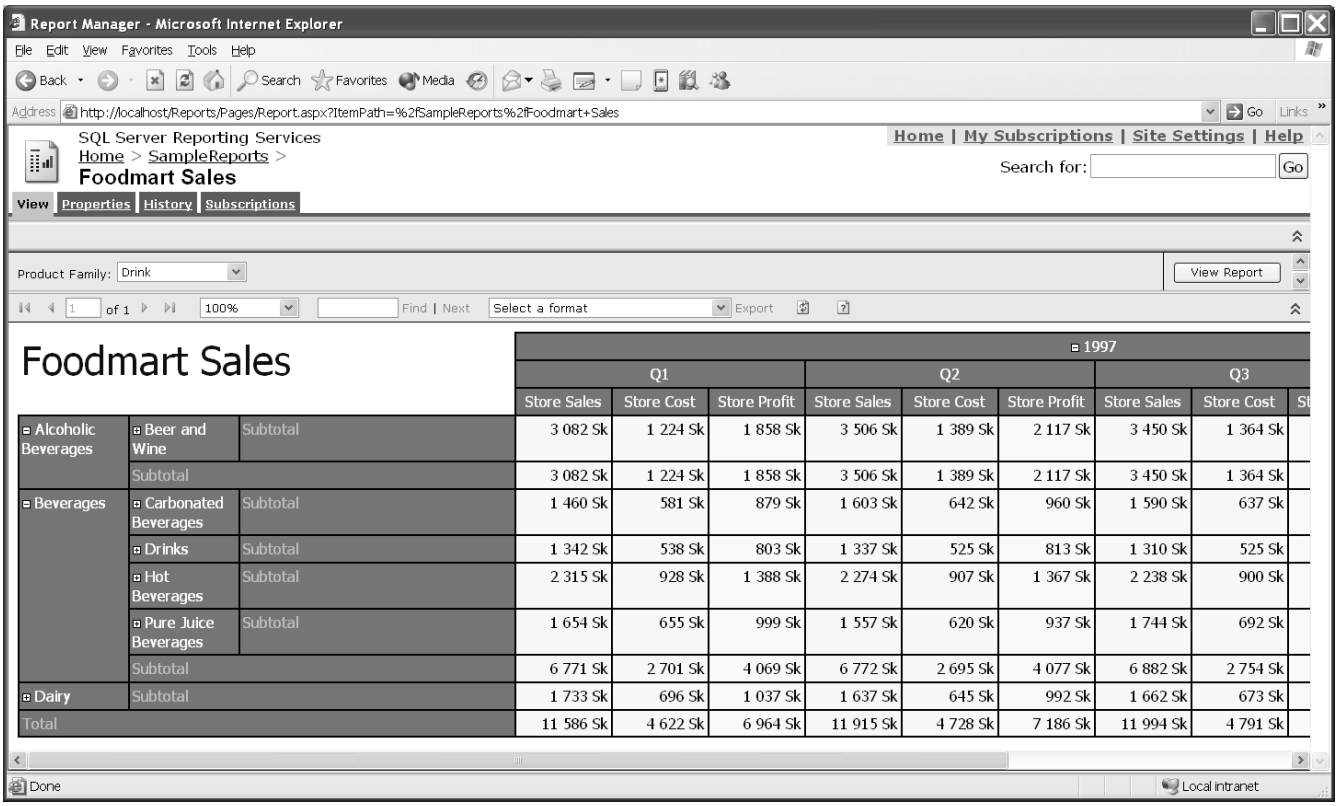
Obr. 3.95 – OLAP report – zložka Layout



Obr. 3.96 – OLAP report – zložka Preview

Všimnime si aj príklad Foodmart Sales, dodávaný spolu s reportovacími službami ktorý je veľmi názornou ukážkou návrhu reportu nad analytickými údajmi. MDX dopyt použitý v príklade je

```
SELECT { [Measures].[Store Sales], [Measures].[Store Cost]  } ON COLUMNS,
{  Descendants([Product].[All Products], [Product].[Brand Name], LEAVES)  } ON ROWS,
{ Time.[1997].[Q1],Time.[1997].[Q2],Time.[1997].[Q3],Time.[1997].[Q4] } ON PAGES
FROM Sales
```



Obr. 3.97 – OLAP report – zložka Preview

Kapitola 4:

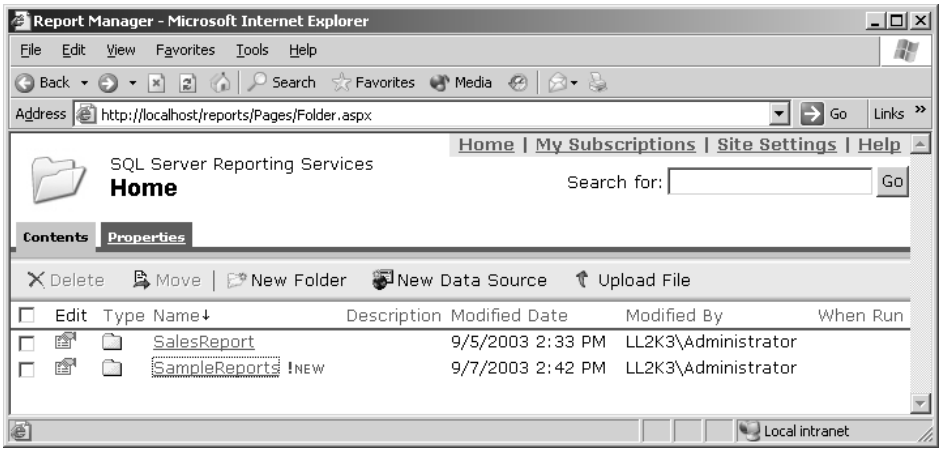
Reportovacie služby z pohľadu administrátora

Základnou ideovou osou tejto publikácie je životný cyklus reportu, ktorý pozostáva z troch fáz

- Návrh
- **Správa**
- Doručenie

Táto kapitola je venovaná fáze správy reportu. Hlavnou „domácou“ stránkou reportovacieho servera, konkrétne jeho časti Report Managera, ktorý slúži pre správu reportov je (na lokálnom počítači) stránka s URL adresou

http://localhost/reports



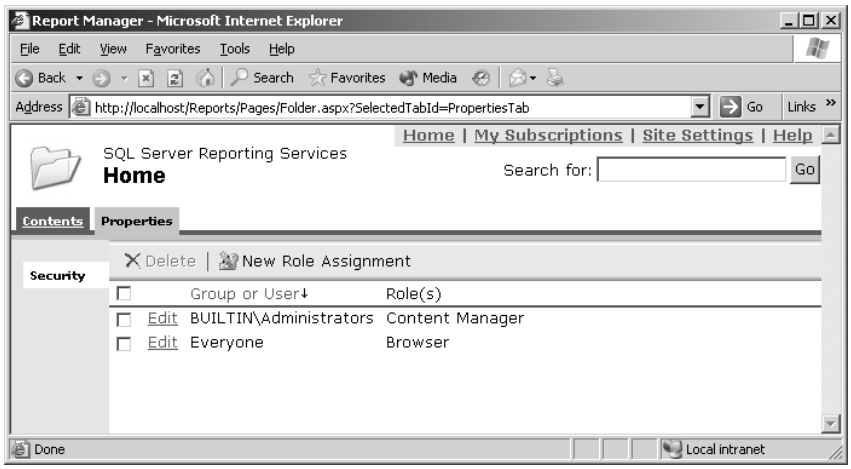
Obr. 4.1 Hlavná stránka Report Managera (záložka Contents)

Na tejto stránke je prehľadne usporiadané niekoľko skupín ovládacích prvkov. Vpravo úplne hore je malý textový toolbar s položkami

- Home
- My Subscriptions
- Site Settings
- Help

Pod ním sa nachádza hlavná pracovná plocha vývojového prostredia s vlastným „ikonovým“ toolbarom.
Než sa k nemu ale dostaneme, všimnime si, že táto plocha pozostáva z dvoch záložiek

- Contents
- Properties



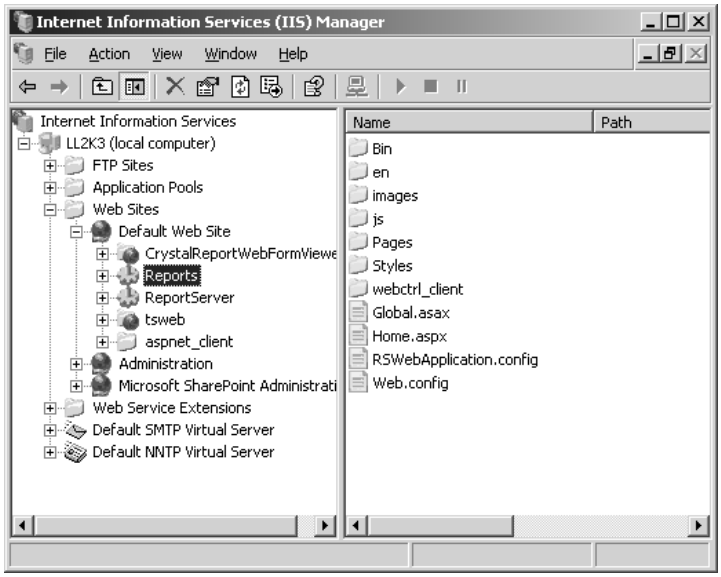
Obr. 4.2 Hlavná stránka Report Managera (záložka Properties)

pre hĺbavejších čitateľov ešte vysvetlíme, čo znamenajú URL adresy

http://localhost/reports
http://localhost/reportserver

Ak si spustíme administrátorský nástroj IIS Manager (je v zložke administrátorských nástrojov operačného systému dostupnej cez menu START – Control Panel), zistíme, že virtuálnym adresárom reports a reportserver zodpovedajú fyzické adresáre:

reports	C:\Program Files\Microsoft SQL Server\MSSQL\Reporting Services\app
reportserver	C:\Program Files\Microsoft SQL Server\MSSQL\Reporting Services\ReportServer



Obr. 4.3 Virtuálne adresáre Internet Information Servera

K správe reportovacích služieb pristupujeme práve cez toto rozhranie. Pre ďalšie ukážky je dôležité, aby sme mali pod správou reportovacieho servera viac projektov, prípadne aspoň jeden zložitejší projekt. Pre tento účel môžeme využiť cvičné príklady, ktoré sú po implicitnej inštalácii reportovacích nainštalované v adresári

```
C:\Program Files\Microsoft SQL Server\MSSQL\Reporting Services\Samples
```

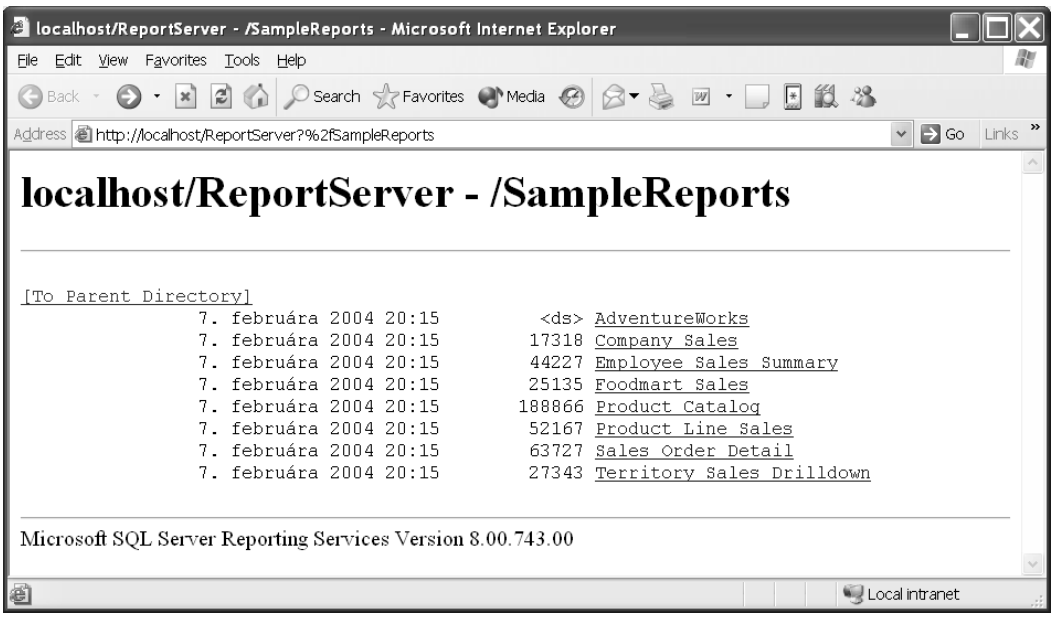
Príklady sú súčasťou jedného riešenia, takže ich môžeme zaviesť do vývojového prostredia Visual Studio .NET 2003 všetky naraz. V prieskumníku súborov stačí dvojklik na súbor

```
SampleReports.sln
```

a celý projekt sa automaticky zavedie do vývojového prostredia. Potom v menu Build aktivujeme funkcie „Build Solution“ a „Deploy Sample Reports“. Ak máme reportovacie služby správne nakonfigurované, mali by sme vo výstupnom okne vývojového prostredia získať výpis o úspešnom preklade a zavedení všetkých príkladov.

```
------ Deploy started: Project: SampleReports, Configuration: Debug ------

Deploying to http://localhost/ReportServer?%2fSampleReports
Deploying data source '/SampleReports/AdventureWorks'.
Deploying report 'Company Sales'.
Deploying report 'Employee Sales Summary'.
Deploying report 'Foodmart Sales'.
Deploying report 'Product Catalog'.
Deploying report 'Product Line Sales'.
Deploying report 'Sales Order Detail'.
```

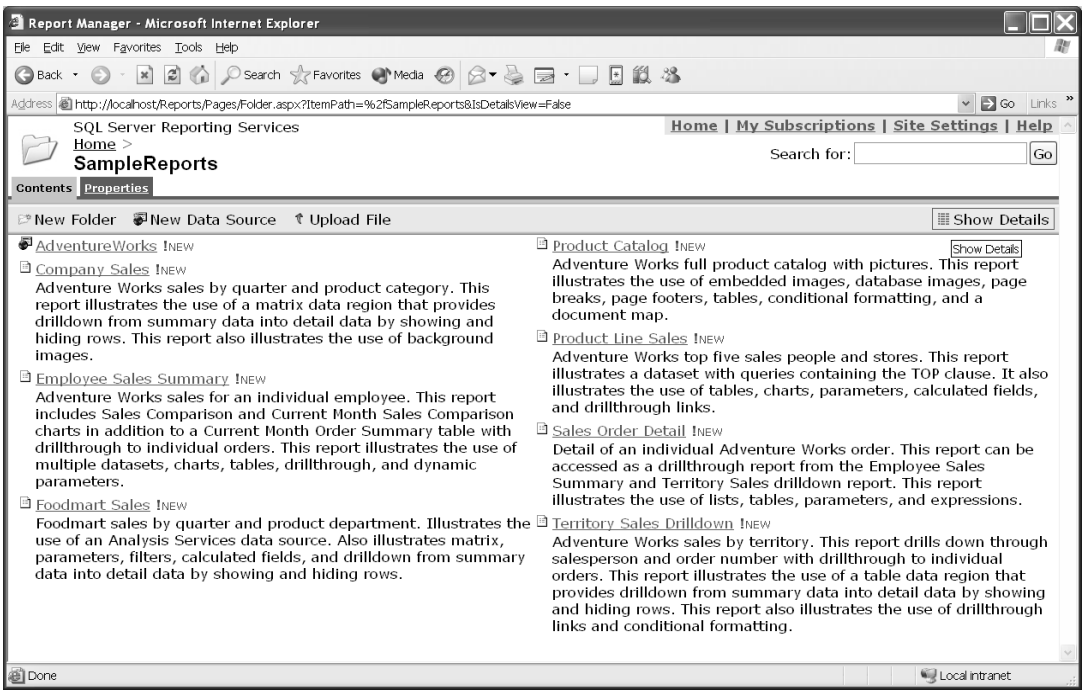
```
Deploying report 'Territory Sales Drilldown'.
Deploy complete -- 0 errors, 0 warnings

---------------------- Done ----------------------

    Build: 1 succeeded, 0 failed, 0 skipped
    Deploy: 1 succeeded, 0 failed, 0 skipped
```



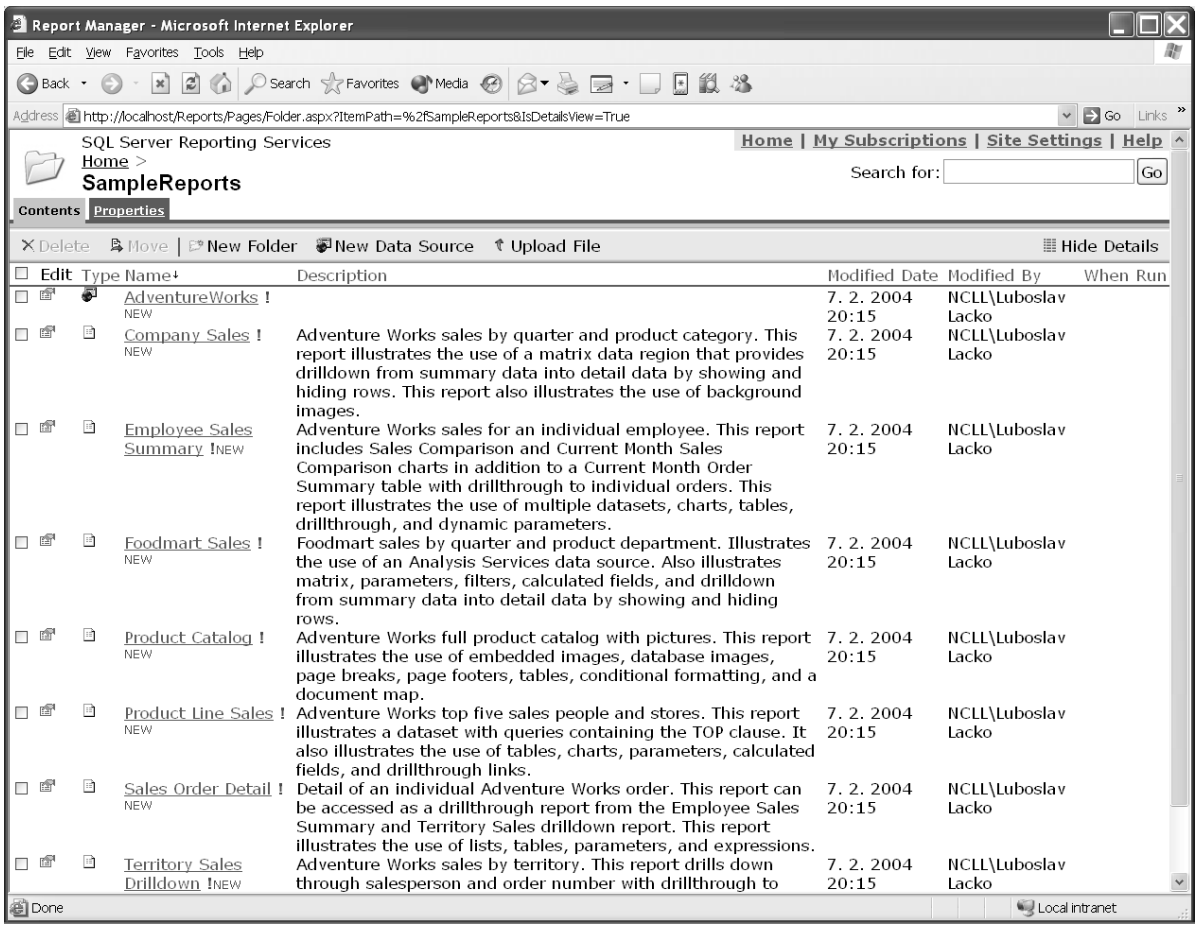
Obr. 4.4 Reporty z projektu Sample Reports

Po zadaní adresy `http://localhost/ReportServer?%2fSampleReports` získame prístup k zoznamu reportov.

Komfortnejší prístup s podrobnejším popisom získame na adrese `http://localhost/Reports` v zložke Sample Reports. Ak zvolíme režim so skrytými detailami, získame prehľadným spôsobom vypísané základné informácie.



Obr. 4.5 Jednotlivé reporty z projektu Sample Reports

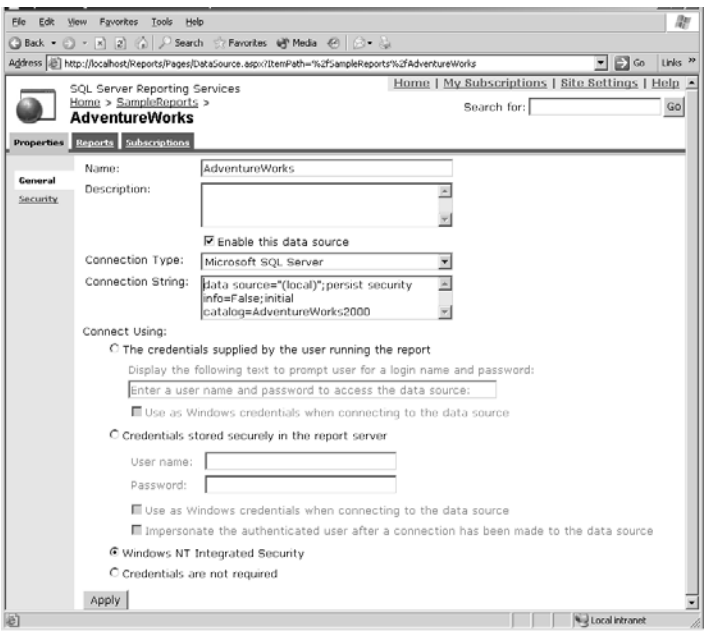


Obr. 4.6 Jednotlivé reporty z projektu Sample Reports s rozvinutými detailami

V podrobnejšom výpise môžeme podľa ikony v stĺpce „Type“ rozlíšiť, či sa jedná o dátový zdroj (prvá položka na našom obrázku), prípadne report.

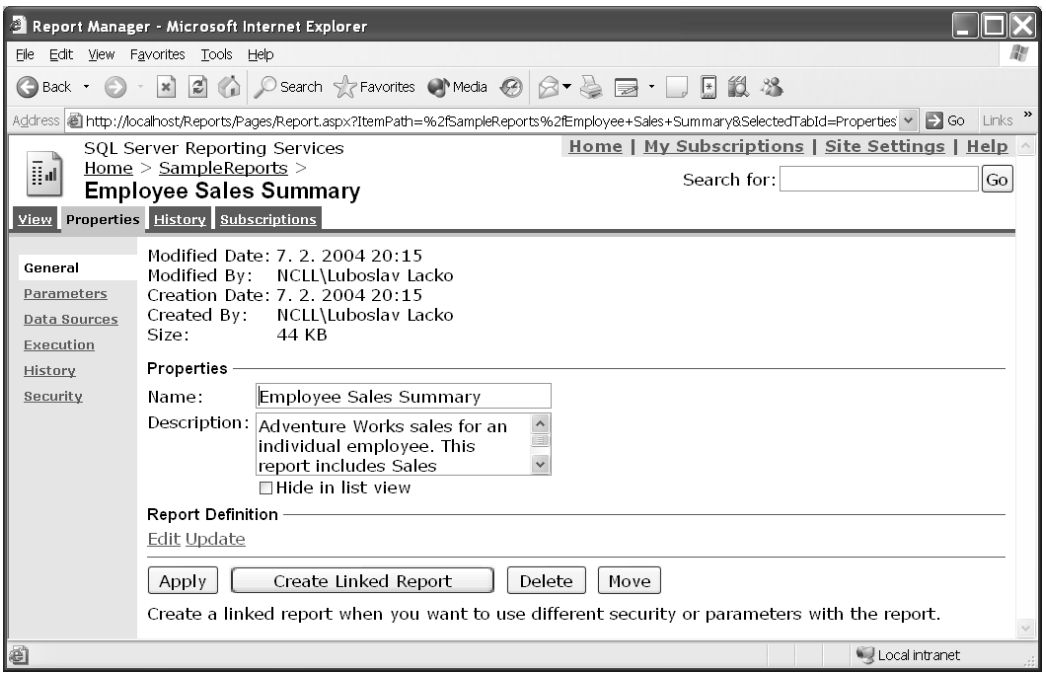
Pre dátový zdroj môžeme zmeniť jeho pripojovací reťazec, prípadne dátový zdroj povoliť, alebo zakázať.

```
data source="(local)";persist security info=False;initial catalog=AdventureWorks2000
```



Obr. 4.7 Správa dátového zdroja AdventureWorks

Od správy dátových zdrojov sa dostávame k správe reportov. Ak klikneme na názov reportu, zobrazí sa nám jeho preview, čiže uvidíme prakticky to isté ako v režime Preview vo Visual Studiu .NET 2003. Kliknutím na ikonu v stĺpci Edit sa dostávame na stánku správy reportu. V našom výklade sa zameriame na report Employee Sales Summary, ktorý je súčasťou cvičných príkladov dodávaných s reportovacími službami



Obr. 4.8 Správa reportu Employee Sales Summary, zložka General

Na ľavej strane si môžeme všimnúť, že každá stránka pre správu reportu má menu zložené z 5, prípadne 6 záložiek (záložka Parameters sa zobrazí len u parametrických reportov)

- General
- Parameters
- Data Sources
- Execution
- History
- Security

Jednotlivé záložky menu správy reportu si popíšeme podrobnejšie.

General

Nosnou časťou tejto záložky je názov reportu a súhrnné informácie o ňom. Pre report Employee Sales Summary sú tieto informácie napríklad v tvare

```
Modified Date:  7. 2. 2004 20:15
Modified By:  NCLL\Luboslav Lacko
Creation Date:  7. 2. 2004 20:15
Created By:  NCLL\Luboslav Lacko
Size:  44 KB
```

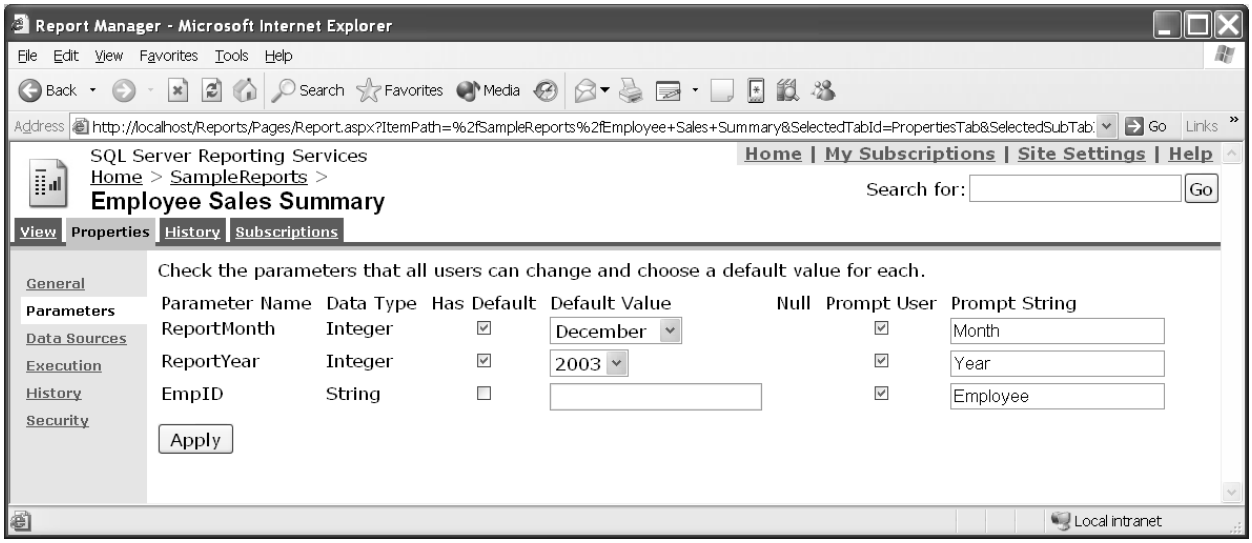
Stránka obsahuje ďalej stručný popis reportu.

```
Adventure Works sales for an individual employee. This report includes Sales Comparison
and Current Month Sales Comparison charts in addition to a Current Month Order Summary
table with drillthrough to individual orders. This report illustrates the use of multiple
datasets, charts, tables, drillthrough, and dynamic parameters.
```

V tomto prípade veľa vyrozumieme aj zo samotného názvu, no pri názve napríklad Report17 pravdepodobne po čase už ani jeho autor nebude mať tušenie aké informácie tento report obsahuje

Parameters

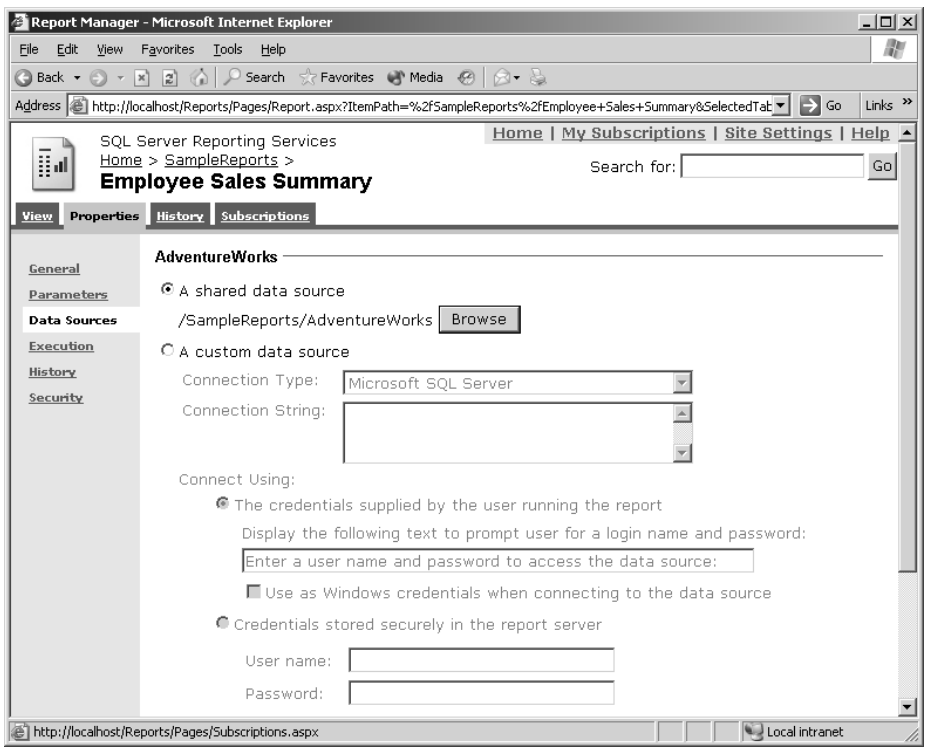
V záložke Parameters je nastavenie parametrov, pre reporty, ktoré tieto parametre vyžadujú. Niektoré parametre môžeme prednastaviť na nejaké implicitné – často používané hodnoty, napríklad časový interval na fiškálne obdobie, môžeme nastaviť aktuálny dátum a podobne. Na stránke sa zobrazí názov parametra a jeho dátový typ. Pomocou checkboxu „Has Default“ môžeme špecifikovať, či predmetný parameter bude mať nejakú implicitne nastavenú hodnotu. Pomocou checkboxu „Prompt User“ naproti tomu stanovujeme, či používateľ bude mať možnosť túto hodnotu nastaviť. V poslednom stĺpci je popis, teda názov parametra, ktorý za zobrazí na stránke reportu vedľa okna pre zadanie hodnoty.



Obr. 4.9 Správa reportu Employee Sales Summary, záložka Parameters

Data Sources

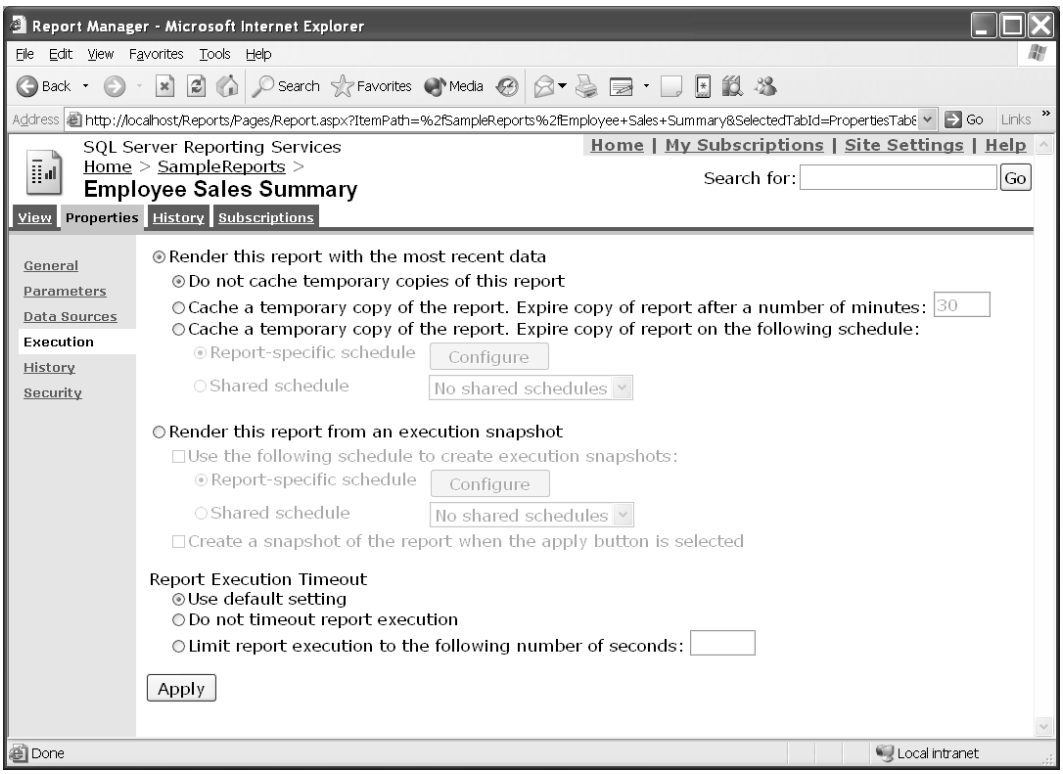
V záložke Data Sources je formulár pre správu a zabezpečenia zdroja údajov. Dátový zdroj môže byť zdieľaný pre viac reportov, v našom prípade databáza Adventure Works, prípadne každý report môže byť napojený na svoj vlastný dátový zdroj



Obr. 4.10 Správa reportu Employee Sales Summary, záložka Data Sources

Execution

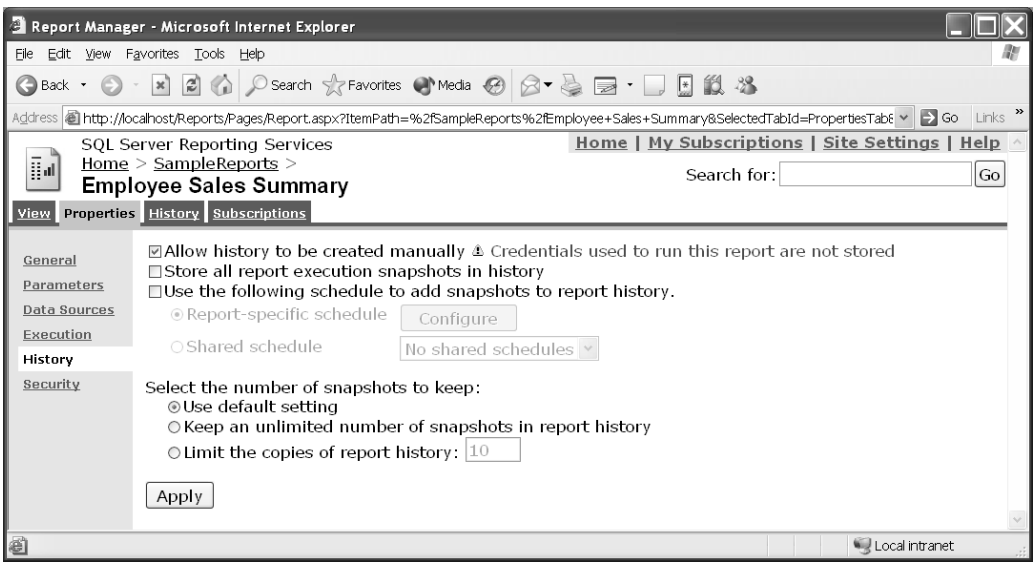
Nastavenia v záložke Execution do značnej miery určujú „fungovanie“ reportu, teda to, akým spôsobom bude využívať kešovanie a časový rozvrh generovania reportu. Idea kešovania je jednoduchá. Pri prvej požiadavke (prípadne automaticky na základe časového rozvrhu) sa vygeneruje report. Prístupom klienta, ktorý si ho vyžiadal, sa však životná púť reportu spravidla nekončí. Je vysoko pravdepodobné, že ten istý report si v krátkom čase vyžiada iný klient. Preto ho na určitú dobu uložíme do dočasnej pamäte typu cache. Po určitom čase sa report stane neaktuálnym a z pamäte cache môže byť odstránený, prípadne nahradený novým aktuálnym reportom toho istého typu. Ukladanie do cache funguje korektne aj pre parametrizované reporty. Finta je jednoduchá, cache sa vytvára pre každú kombináciu parametrov.



Obr. 4.11 Správa reportu Employee Sales Summary, záložka Execution

History

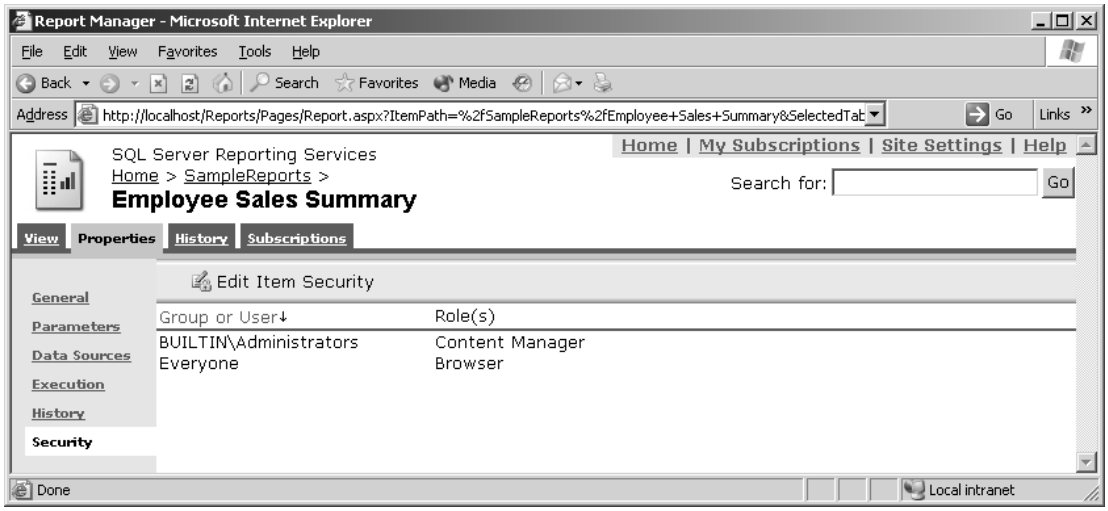
Ukladanie histórie reportu nie je principiálne zložité. Stačí ukladať časové snímky (snapshot) jednotlivých reportov, ktoré sú práve aktuálne. V záložke History definujeme spôsob generovanie snapshotov a počet snímkov do minulosti, ktoré budú pre daný report uchované.



Obr. 4.12 Správa reportu Employee Sales Summary, záložka History

Security

Záložka Security slúži pre zabezpečenie údajov v reporte. Podrobnejšie na túto tému pojednáva stať Bezpečnosť.



Obr. 4.13 Správa reportu Employee Sales Summary, záložka Security

Reporty „na objednávku“

Po vytvorení reportu predpokladáme, že nám bude nejaký čas slúžiť, to znamená, že k nemu budeme nejakým spôsobom pristupovať. Zjednodušene by sme mohli vytvoriť dve principiálne cesty, ako sa report dostane ku klientovi
– od klienta k reportu (klient pristúpi napríklad na URL adresu stránky reportu)
– od reportu ku klientovi (reporty chodia klientovi elektronickou poštou)

V tejto stati sa budeme venovať druhej variante, teda doručovaniu reportov klientovi elektronickou poštou.

Ako predzvesť k tejto funkcionalite je potrebné pri inštalácii nastaviť v príslušnom dialógu adresu SMTP servera a odchodziu mailovú adresu. Ak príslušné parametre pri inštalácií nezadáme a budeme chcieť posielať reporty elektronickou poštou, budeme musieť tieto parametre nastaviť v súbore

RSReportServer.config

ktorý nájdeme v adresári

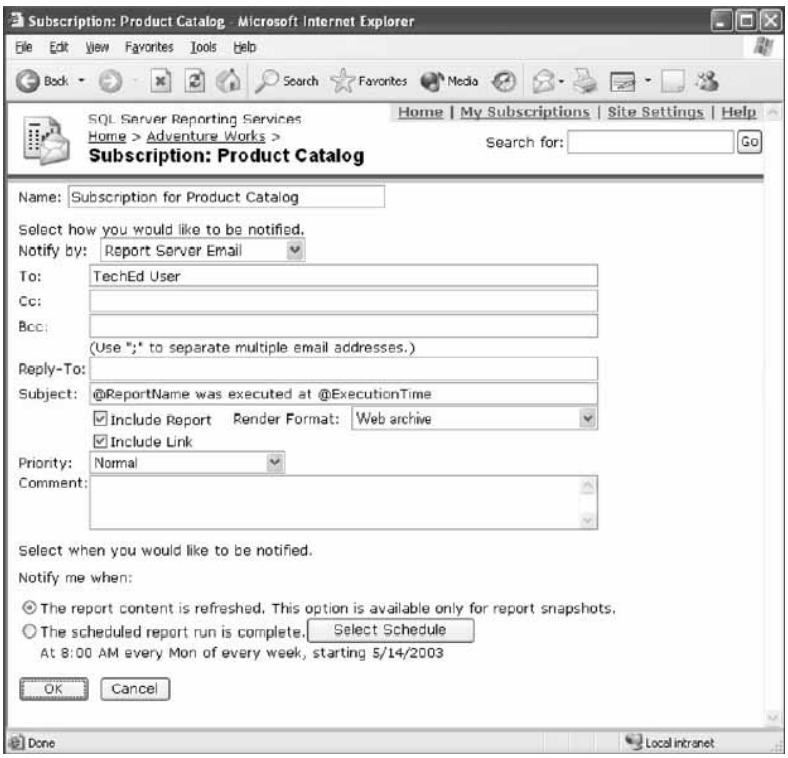
C:\Program Files\Microsoft SQL Server\MSSQL\Reporting Services\ReportServer

```
<Configuration>
    <RSEmailDPConfiguration>
      <SMTPServer></SMTPServer>
      <SMTPServerPort></SMTPServerPort>
      <SMTPAccountName></SMTPAccountName>
      <SMTPConnectionTimeout></SMTPConnectionTimeout>
      <SMTPServerPickupDirectory></SMTPServerPickupDirectory>
      <SMTPUseSSL></SMTPUseSSL>
      <SendUsing></SendUsing>
      <SMTPAuthenticate></SMTPAuthenticate>
      <From></From>
      <EmbeddedRenderFormats>
            <RenderingExtension>MHTML</RenderingExtension>
      </EmbeddedRenderFormats>

      <PrivilegedUserRenderFormats></PrivilegedUserRenderFormats>
```

```
        <ExcludedRenderFormats>
            <RenderingExtension>HTMLOWC</RenderingExtension>
            <RenderingExtension>NULL</RenderingExtension>
        </ExcludedRenderFormats>

          <SendEmailToUserAlias>True</SendEmailToUserAlias>
          <DefaultHostName></DefaultHostName>
          <PermittedHosts></PermittedHosts>
      </RSEmailDPConfiguration>
</Configuration>
```

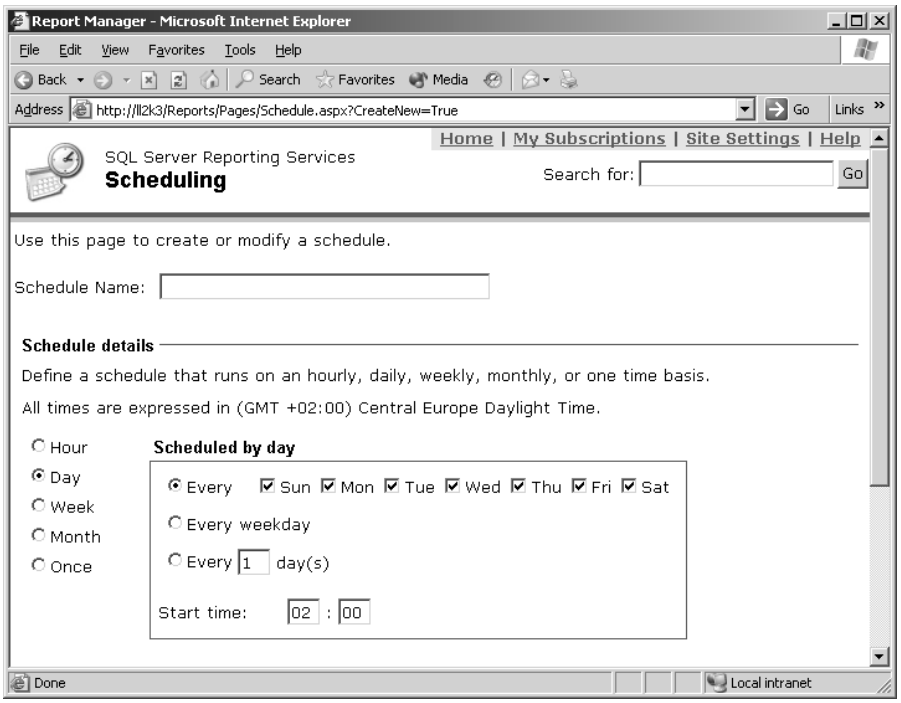
Vidíme, že v tomto prípade sa lenivosť pri inštalácii príliš nevyplatila, je potrebné zadať nielen SMTP adresu, ale aj niekoľko ďalších parametrov. Našťastie je podrobný postup popísaný v nápovede v stati Configuring a Report Server for E-Mail Delivery

Zvyšné parametre nastavíme na stránke nastavenia reportu v zložke „Subscriptions“.
V následne zobrazenom dialógu nastavíme potrebné parametre pre doručovanie reportov.



Obr. 4.14 Subscription

Zatiaľ sme nastavovali, ktoré ako a kam budú reporty doručované. Zostáva nastaviť kedy a ako často budú jednotliví klienti reporty dostávať. Tieto informácie nastavíme na stránke „Scheduling“



Obr. 4.15 Nastavenie časového harmonogramu doručovania reportu

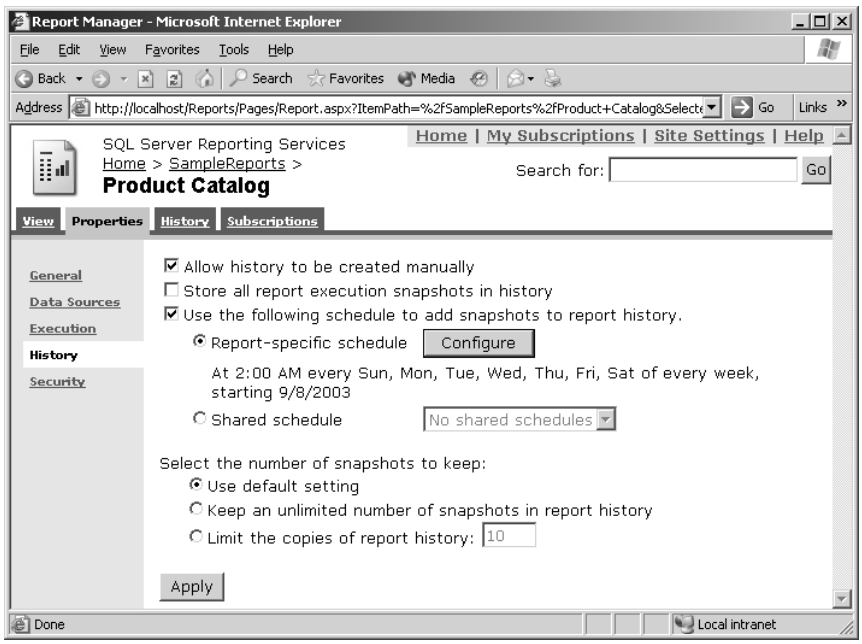
Pokiaľ to potrebujeme, môžeme nastaviť uschovávanie statických kópií reportu a takto dokumentovať jeho históriu. Na stránke pre nastavenie parametrov reportu klikneme v menu v ľavej časti stránky na zložku History.

V zložke history nastavíme požadovaný režim, My sme nastavili voľbu

„Use the following schedule to add snapshots to report history“

a pomocou podobného dialógu ako pre doručovanie reportu sme nastavili časový harmonogram. Ponechali sme implicitnú voľbu, teda

„At 2:00 AM every Sun, Mon, Tue, Wed, Thu, Fri, Sat of every week, starting 9/8/2003“



Obr. 4.16 Subscription

Data – Driven Subscription

Aj keď v mnohých prípadoch sú informácie tovarom a anglický pojem „subscription“ teda „predplatné“ je pri dodávaní reportov mimo organizáciu celkom na mieste, budeme pojem „subscription“ voľne prekladať ako objednávku. Reporty sa teda doručujú odberateľom na základe objednávky a to buď v zadanom čase, alebo ako reakcia na nejakú udalosť. Zatiaľ čo pri štandardnom doručovaní reportov boli objednávky vytvárané a riadené pre individuálnych používateľov, Data – driven subscription získava údaje o spôsobe doručovania z vlastnej databázy, kde má aktuálne informácie o odberateľoch. Data – driven subscription môžeme použiť len vo verzii Enterprise

Pre ilustráciu ukážeme postup vytvorenia databázy odberateľov pre data – driven subscription. Použijeme štandardne s SQL Serverom dodávanú konzolovú aplikáciu SQL Query Analyzer

Vytvoríme novú databázu *Subscribers* pod správou SQL Servera 2000.

```
CREATE DATABASE Subscriber;
```

Prepneme konzolovú aplikáciu do novovytvorenej databázy príkazom

```
USE Subscriber;
```

Vytvoríme tabuľku UserInfo, ktorá bude obsahovať stĺpce *Name, Alias, EmployeeID, Format, Linked*
Všetky stĺpce môžu byť dátového typu `varchar(50)`

```
CREATE TABLE UserInfo
(
   Name         varchar(50),
   Alias        varchar(50),
   EmployeeID   varchar(50),
   Format       varchar(50),
   Linked       varchar(50)
);
```

Do tabuľky UserInfo povkladáme záznamy o odberateľoch reportov, napríklad

```
INSERT INTO UserInfo (Name, Alias, EmployeeID, Format, Linked)
  VALUES ('Fernando Caro', 'fc@company.com', '24', 'IMAGE', 'True');

INSERT INTO UserInfo (Name, Alias, EmployeeID, Format, Linked)
  VALUES ('Rachel Valdez', 'rv@company.com', '35', 'MHTML', 'True');

INSERT INTO UserInfo (Name, Alias, EmployeeID, Format, Linked)
 VALUES ('Michael Blythe', 'mb@company.com', '38', 'PDF', 'False');
```

Ak sa chceme presvedčiť o úspešnosti dosiaľ vykonaných krokov necháme si vypísať obsah tabuľky UserInfo

```
SELECT * FROM UserInfo;

Name                     Alias            EmployeeID         Format      Linked
-----------------------  ---------------  -----------------  ----------  ----------
Fernando Caro            fc@company.com    24                 IMAGE       True
Rachel Valdez            rv@company.com    35                 MHTML       True
Michael Blythe           mb@company.com    38                 PDF         False

(3 row(s) affected)
```

Následne špecifikujeme uložené prístupové privilégiá. V okne Report Managera vyberieme report **Employee Sales Summary** a v záložke **Properties** vyberieme položku **Data Sources**. Označíme voľbu **A Custom data source**.

Po označení tejto voľby sa nám sprístupnia parametre pripojenia, ktoré vyplníme. Typ pripojenia bude MS SQL Server

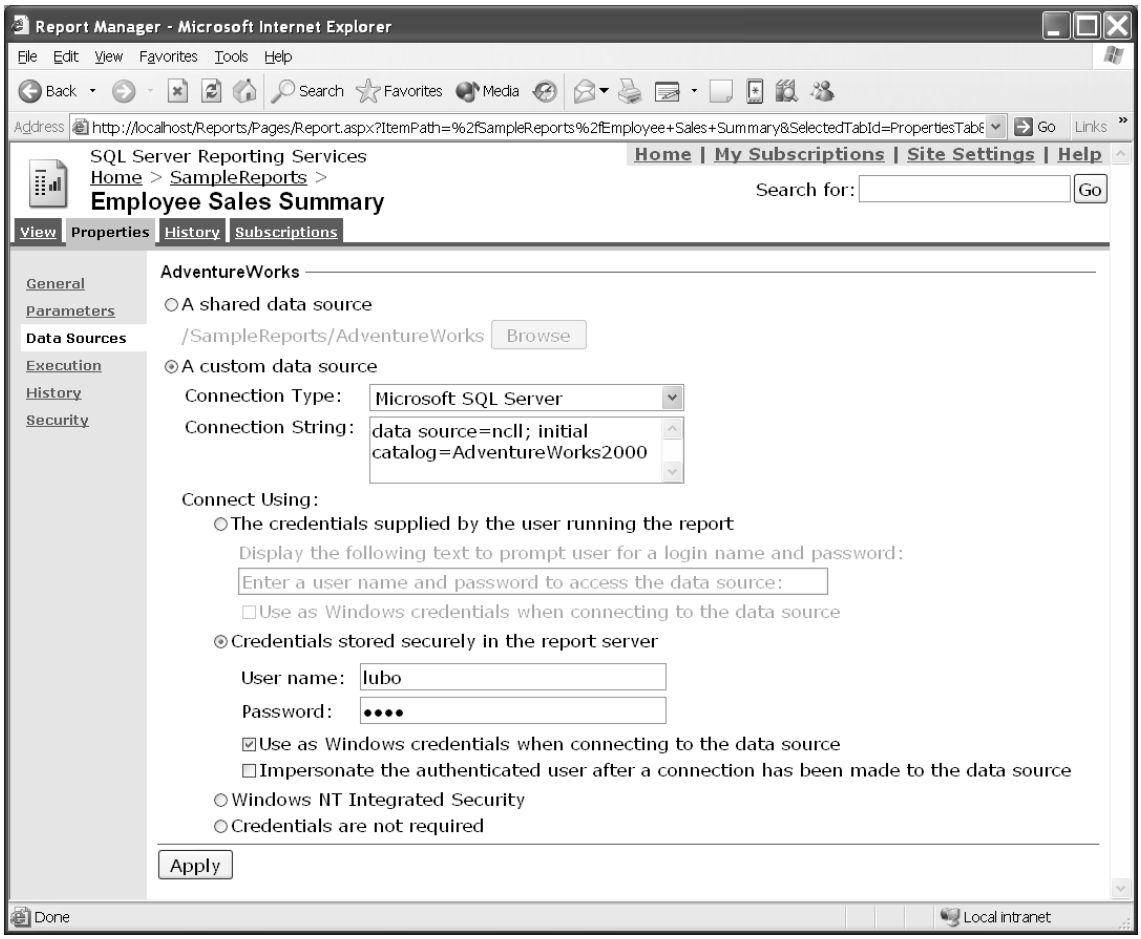
```
data source=<názov vášho servera>; initial catalog=AdventureWorks2000
```

V našom prípade

```
data source=ncll; initial catalog=AdventureWorks2000
```

Zaškrtnutím voľby **Credentials stored securely in the report server** sprístupníme dialóg pre zadanie mena a hesla. Taktiež zaškrtneme voľbu **Use as windows credentials when connecting to the data source**

Prepnutím do záložky View sa presvedčíme o správnosti prístupových parametrov



Obr. 4.17 Nastavenie prístupových privilégii pre data – driven subscription

Aby sme mohli vytvoriť nový predpis pre doručovanie, v záložke Subscription aktivujeme položku New Data – driven Subscription.

Obr. 4.18 Nastavenie prístupových privilégii pre data – driven subscription

Ako delivery method vyberieme Report Server Email a následne nastavíme parametre pre Subscriber Data Source. Napokon nastavíme parametre pre doručovanie reportov.

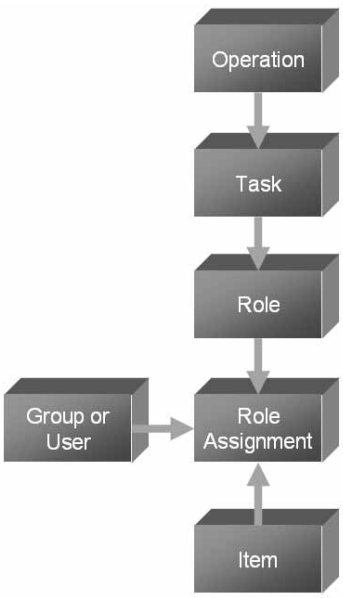
Bezpečnosť

Táto stať bude akousi nadstavbou, ktorá popíše zabezpečenie reportov. Základom je totiž zabezpečenie samotných údajov. Zamyslime sa najskôr nad tým, prečo je vlastne dôležitá administrácia prístupu k databázam a aké údaje tieto databázy spravidla obsahujú. Transakčné databázy sú typickým príkladom viac používateľského prostredia. Preto je veľmi dôležité definovať prístupové práva jednotlivých používateľov k databázovým objektom a špecifikovať rozsah ich oprávnení. Niektorí používatelia môžu údaje do databázy zapisovať, prípadne ich vymazávať, iní majú sprístupnené len čítanie údajov. Používateľský prístup má na starosti správca databázy väčšinou v spolupráci so správcom systému. Ak by sme mali zhrnúť obsah tohoto odstavca jednoducho je potrebné zabrániť prístupu neoprávnených osôb k údajom v databázach.

Ak by sme však mali porovnať „citlivosť“ a obchodnú hodnotu údajov uložených v transakčných databázach so súhrnnými informáciami v reportoch a v neposlednom rade aj v analytických databázach a reportoch nad analytickými databázami rozdiel je priepastný. Ak by sa aj niekomu podarilo „ukoristiť“ niekoľko tabuliek alebo aj celú transakčnú databázu úžitok by z toho mal len po zložitom vyhodnocovaní. Kdežto po získaní prístupu k reportu má narušiteľ všetky informácie naservírované ako na striebornej tácke.

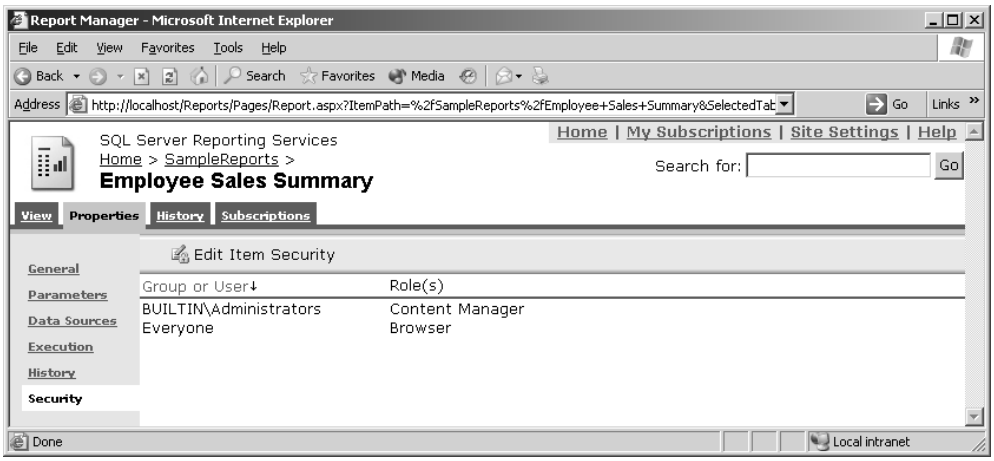
Bezpečnostný model reportovacích služieb je založený na roliach. Ak začneme vytvárať používateľov pre zložitejší projekt a prideľovať im prístupové práva, čoskoro prídeme na to, že sa nám vytvárajú skupinky používateľov, ktoré majú tieto práva rovnaké. Je to analógia s bežnou hierarchiou vo firmách, kde pracuje viac pracovníkov v rovnakom pracovnom zaradení. Preto sa pri administrovaní databáz používajú role. Role umožňujú združovať používateľov do skupín. Medzi používateľmi a rolami je vzťah M:N, to znamená, že nielen viacero používateľov je priradených k jednej role, ale aj opačne, jeden používateľ môže mať viac rolí. V reportovacích službách SQL serveru môžeme definovať dva typy rolí

- role pre prístup k správe reportovacích služieb
- role pre prístup ku konkrétnym reportom



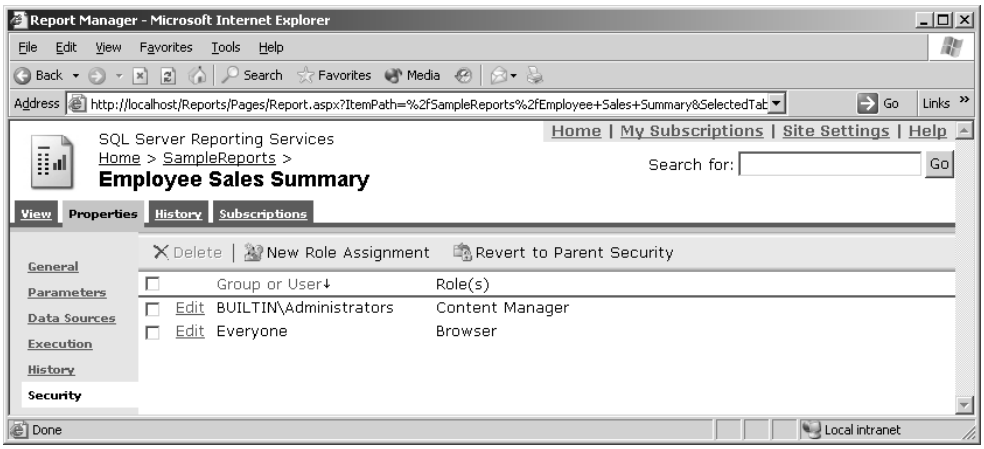
Obr. 4.19 Bezpečnostný model založený na roliach

V predchádzajúcej stati sme stručne poukázali na existenciu záložky Security



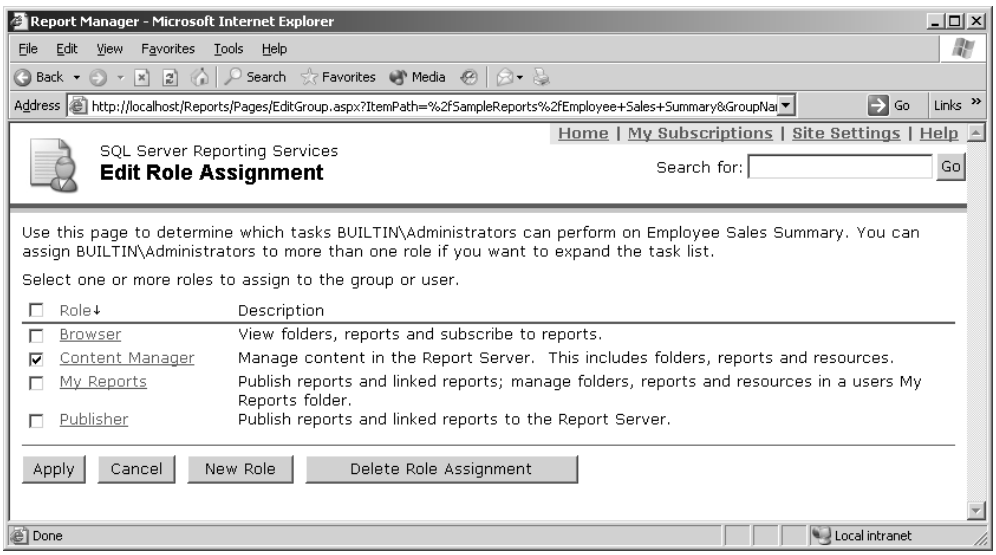
Obr. 4.20 Správa reportu Employee Sales Summary, záložka Security

Všimnime si tlačidla „Edit Item Security“. Po jeho zatlačení prechádzame z celkového zabezpečenia projektu na detailnú úroveň zabezpečenie. Naspäť sa dostaneme tlačidlom „Revert to Parent Security“. Na detailnej úrovni zabezpečenia môžeme definovať a editovať nastavenia pre jednotlivých používateľov a pre role.



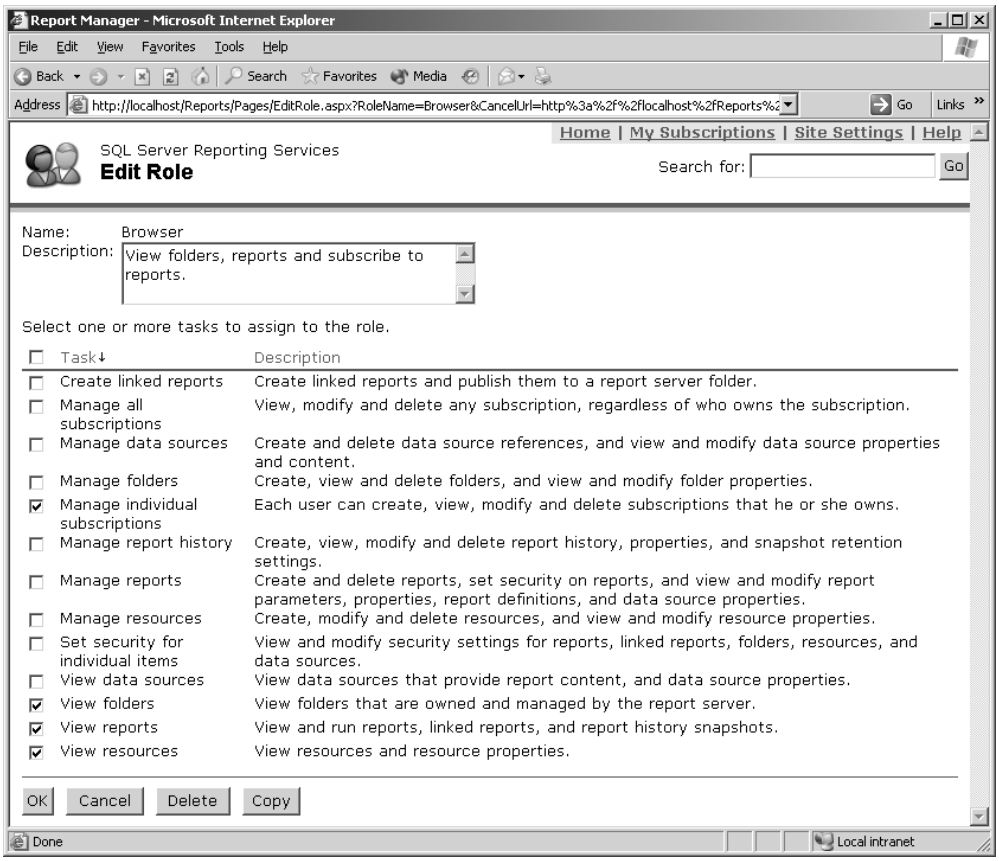
Obr. 4.21 Správa reportu Employee Sales Summary, záložka Security

Zobrazí sa formulár, kde sú jednotlivým používateľom pridelené príslušné role. Nastavenie môžeme zmeniť pomocou linku „Edit“. V následne zobrazenom formulári vidíme ktoré role môžeme prideliť predmetnému používateľovi.



Obr. 4.22 Prideľovanie rolí používateľom

Role definujeme pomocou formulára, v ktorom pomocou zaškrtávacích políčok povolíme, prípadne nepovolíme príslušnú činnosť.



Obr. 4.23 Definovanie rolí

Prístup k reportom je možný pomocou rozhrania WSDL ako ku webovej službe. Po zadaní URL adresy

http://localhost/reportserver

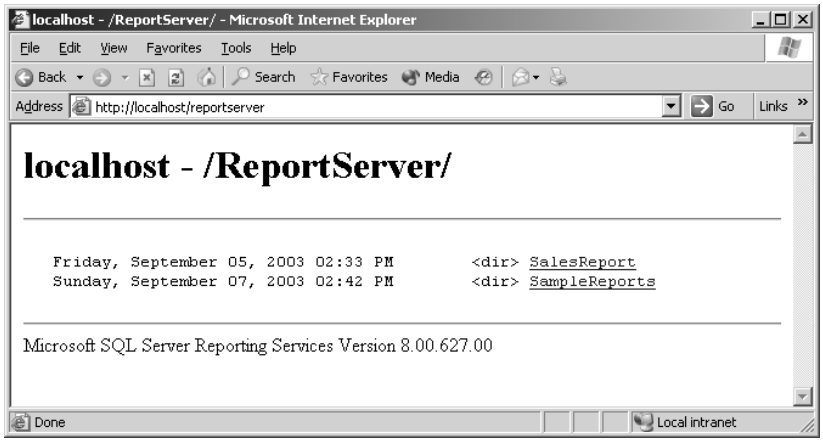
pristúpime na stránku webovej služby

http://localhost/reportserver/ReportService.asmx

Táto stránka je technicky vzatá len odkazom na binárny kód (napísaný v jazyku C#)

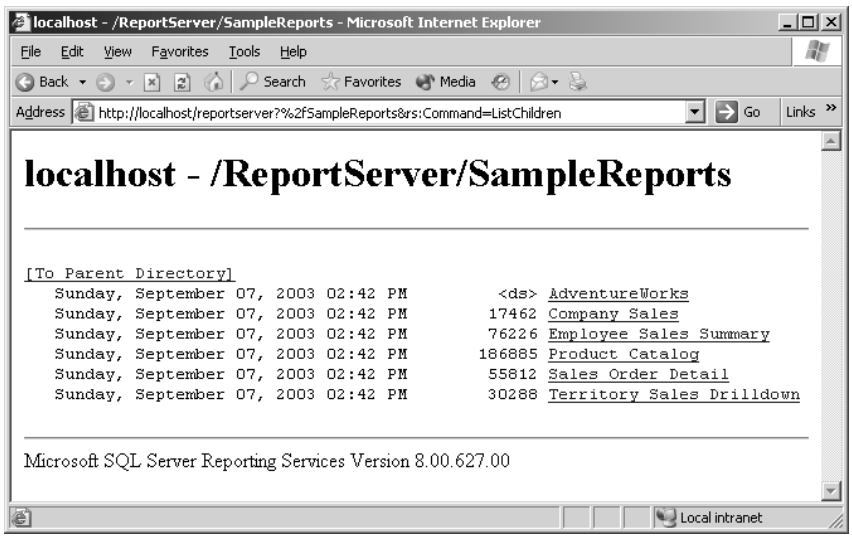
```
<%@ WebService Language="c#" Codebehind="ReportService.asmx.cs"
Class="Microsoft.ReportingServices.WebServer.RSWebService" %>
```

Rozhranie webovej služby nám umožňuje prístup k jednotlivým projektom (adresárom)



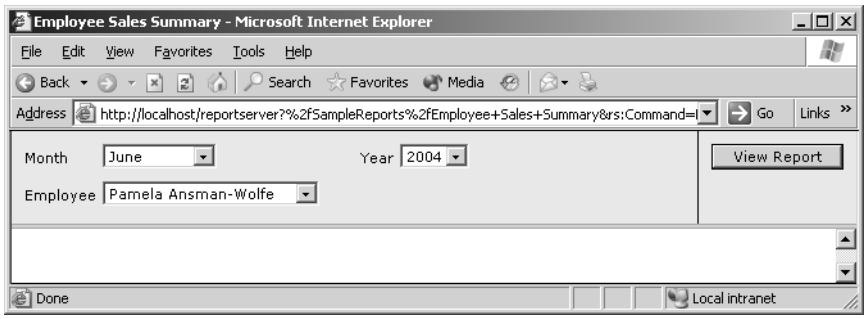
Obr. 4.24 prístup k reportom cez webovú službu

po vnorení do niektorej úrovne máme prístup k jednotlivým reportom

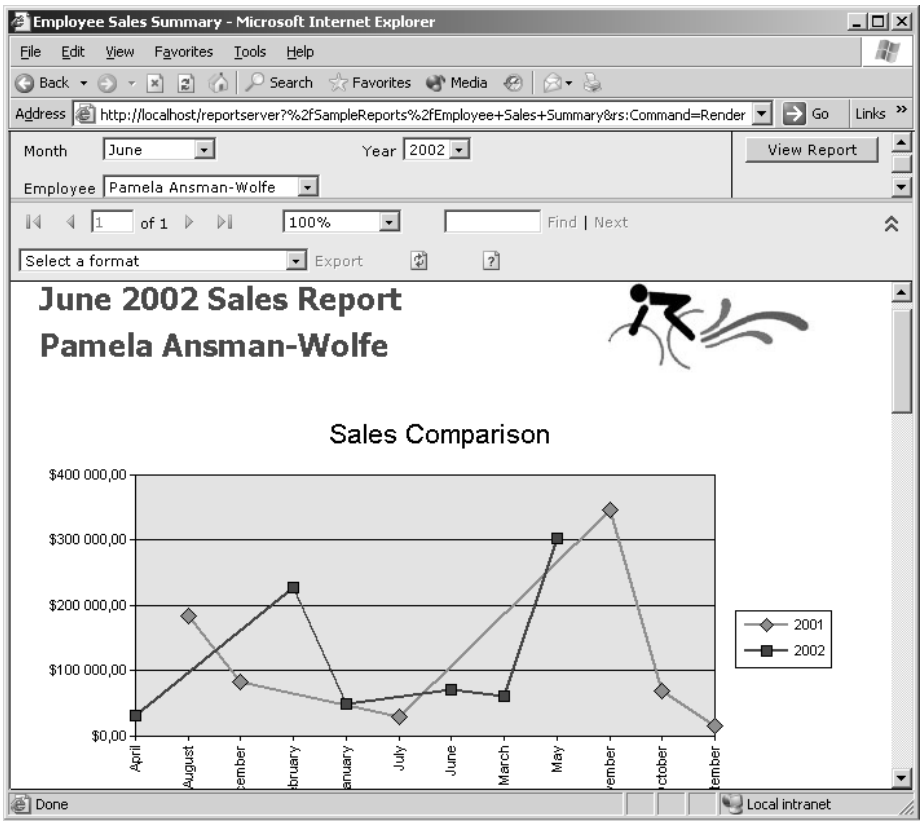


Obr. 4.25 prístup k reportom cez webovú službu

Reporty si môžeme prezerať a pokiaľ sú parametrické, musíme najskôr zadať príslušné parametre.



Obr. 4.26 Zadávanie parametrov parametrického reportu



Obr. 4.27 Zobrazenie reportu pre zadané parametre

Kapitola 5:

Embedding do zákaznických a partnerských riešení

Táto kapitola sa bude venovať fáze doručenia reportu. Termín „doručenie“ je vzhľadom k možnostiam reportovacích služieb obzvlášť v prostrediach .NET, Office, Share Point a rôznych aplikačných serverov značne skromný a zjednodušený. Ale viac menej dostatočne vystihuje, že ide o to, aby sa klient dostal včas k reportu, ktorý vyžaduje a navyše z toho prostredia na ktoré je zvyknutý, s ktorým denne pracuje.

Poľa filozofie doručovania sme už v kapitole o správe predstavili obidva spôsoby:
– doručovanie reportov na vyžiadanie (on-demand, „pull“)
– udalosťami riadené doručovanie (event-based, „push“)

Nezanedbateľnou skutočnosťou je aj formát, v akom bude dokument reportu ku klientovi doručený. K dispozícii sú webové formáty (HTML 4 alebo 3.2), printovateľné formáty (TIFF, PDF) alebo aj dátové formáty (Excel, XML, CSV). Voľba formátu závisí od toho, pre aký účel report potrebujeme, prípadne pomocou akých prehliadačov alebo programov budeme s reportom pracovať.

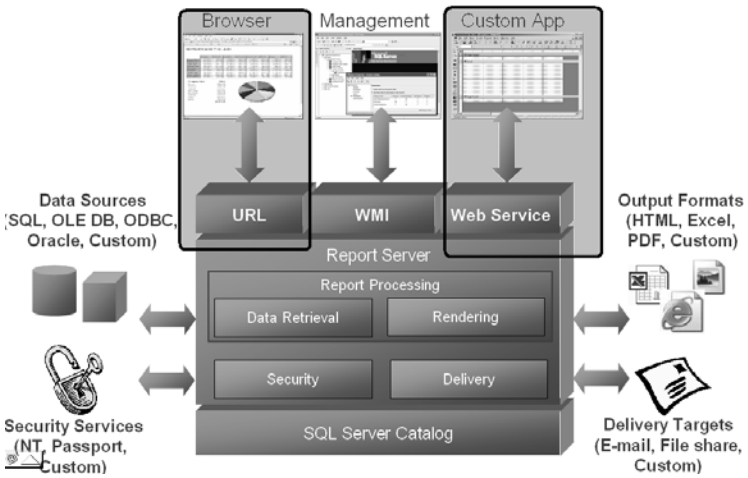
Ešte raz je potrebné zdôrazniť myšlienku z predchádzajúceho odstavca o tom, že je potrebné, aby klient dostával reporty (presnejšie by sme sa mohli vyjadriť aby mu reporty boli zobrazované) v tom aplikačnom prostredí, v ktorom denne pracuje, na ktoré je zvyknutý a v ktorom rieši problémy, pred ktoré je postavený. Predpokladáme totiž, že prezeranie reportov nie je samoúčelné, ale slúži na podporu rozhodovania a teda že po analýze reportu bude spravidla nasledovať nejaký zásah do príslušného procesu.

Skôr než začneme preberať jednotlivé varianty vkladania reportov do aplikácií, je potrebné ujasniť si filozofiu použitia reportovacích služieb. Aj keď je možné vyrendrovať XML dokument, ťažisko reportovacích služieb je v zobrazovaní údajov vo forme reportov. Teda našou úlohou bude zjednodušene povedané "vložiť" zobrazovaciu plochu reportu na plochu aplikačného rozhrania aplikácie, alebo technicky vzaté, potrebujeme zobraziť okno prehliadača webového obsahu na ploche aplikácie.

Pre úplnosť zosumarizujeme programátorské rozhrania, pomocou ktorých môžeme prepájať reportovací server s aplikačnou logikou

Definícia reportu – definícia reportu v XML dokumente využívajúca otvorenú schému
Rozhranie pre **prehliadanie reportu** – prístup cez URL adresu alebo cez SOAP protokol webovej služby
Rozhranie pre **správu reportu** – prístup cez WMI alebo cez SOAP protokol webovej služby

Oblasti v ktorých sa budeme pri embeddingu reportov do aplikácií pohybovať sú v schéme architektúry vyznačené šedými obdĺžnikmi.



Obr. 5.1. Vrstva náväznosti na aplikáciu v hierarchickej štruktúre reportovacích služieb

Z hľadiska princípov rozdeľujeme klientov na tenkých a tlustých. Na základe tohoto členenia budeme popisovať aj spôsob embeddingu reportov do jednotlivých typov klientských aplikácií

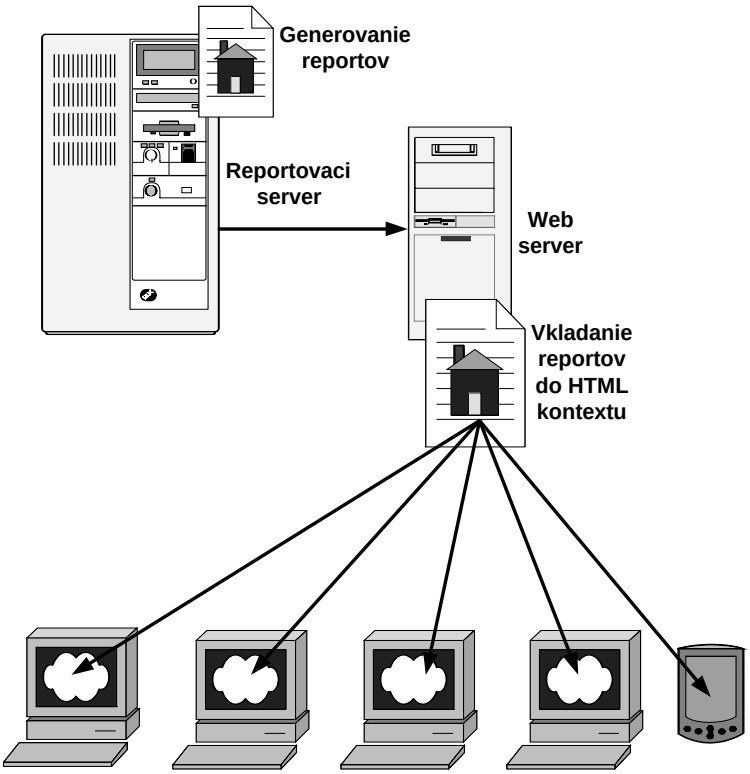
Pod pojmom **tenký klient** rozumieme spravidla počítač, kde sa pre prístup k údajom používa prehliadač webových stránok. To znamená, že celá aplikačná logika beží na serveri a výpočtové možnosti klientského počítača viac – menej

ležia ľadom, využívajú sa hlavne pre zobrazenie. Pod pojmom tenký klient môžeme teda chápať nielen akýkoľvek počítač, od najvýkonnejšieho servera, až po low-endové zostavy, ale aj vreckové počítače triedy Pocket PC, Palm, ba dokonca aj mobilné telefóny s podporou protokolu WAP. Výhodou použitia tenkého klienta je hlavne podpora najširšieho spektra zariadení a platforiem a prístup prostredníctvom prehliadača webových stránok. Hlavnou nevýhodou je, že takéto aplikácie z princípu nepracujú v off - line režime, vo väčšine prípadov málo využívajú výkon lokálneho zariadenia a sú tu aj isté obmedzenia týkajúce sa používateľského rozhrania, ktoré vyplývajú napríklad z toho, že HTTP protokol je bezstavový.

Pod pojmom **tlustý klient** (v anglickej terminológii rich client) budeme rozumieť klientskú aplikáciu, ktorá beží na lokálnom počítači, a teda môže naplno využívať výkon jeho hardvéru a možnosti operačného systému. Pre svoju činnosť ale využíva údaje a službu nejakého servera. Výhodou takéhoto riešenia je, že aplikácie pracujú v obidvoch režimoch off - line aj on - line, pričom si počas pripojenia môže klientský počítač synchronizovať, prípadne replikovať údaje so serverom. Nevýhodou je, že pre každú platformu (Windows. Linux, iMAC, Pocket PC 2002..) musíme aplikáciu vyvíjať zvlášť. Táto nevýhoda sa čiastočne stiera použitím tzv. riadeného (managed) kódu, ktorý je pre viacero platforiem rovnaký. Takto môžeme napríklad vyvíjať aplikácie v programovacích jazykoch Visual Basic a Visual C# pre platformu Windows aj Pocket PC 2002.

Embedding do aplikácií a riešení typu „tenký klient“

Problematiku „vkladania“ reportov môžeme v tomto prípade zúžiť na zobrazenie príslušného reportu (jednoznačne identifikovaného pomocou URL adresy) v okne prehliadača webových stránok. Najčastejšie budeme zrejme zobrazovať reporty pomocou frame, pričom v inom frame, alebo mimo neho môžu buť ostatné informácie, prípadne ovládacie prvky pre nastavenie parametrov zobrazovaných reportov. Čo sa týka architektúry, aplikačný server môže a nemusí byť totožný s webovým, prípadne databázovým serverom. Čiastočne je do dané aj licenčnou politikou pre jednotlivé verzie reportovacích služieb. Samotné pripájanie reportov k aplikáciám sa deje na úrovni web serveru, ktorý generuje definitívny obsah HTML stránok, ktoré sa zobrazia klientovi v okne jeho prehliadača.



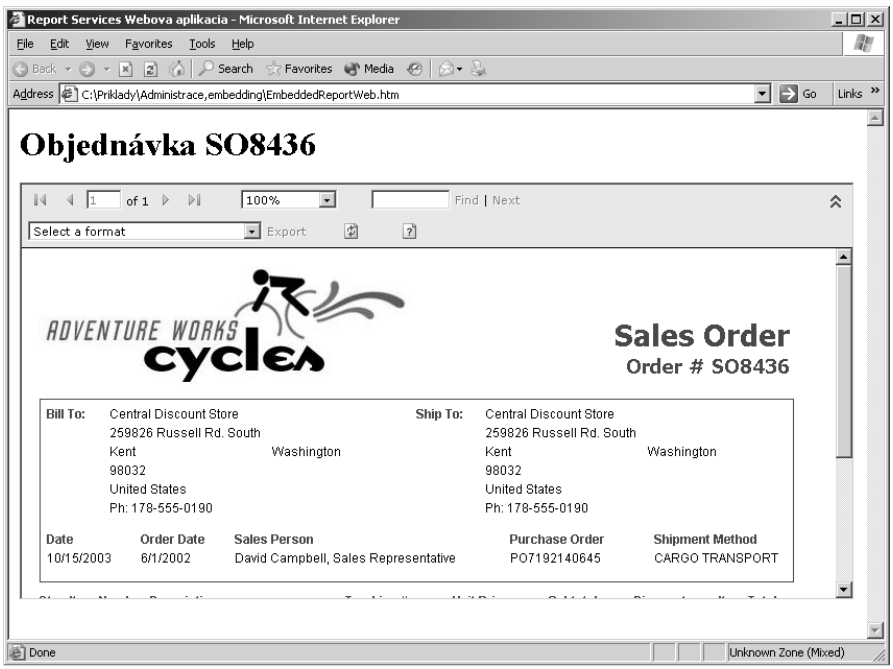
Obr. 5.2. Príklad architektúry pre využívanie reportovacích služieb v aplikácii typu „tenký klient“

Pre vývoj aplikácií môžeme využívať prakticky ľubovoľný programovací jazyk alebo serverový skriptový systém (PHP, ASP, Java, JSP, ASP.NET....), prípadne len HTML alebo DHTML aplikáciu. Princíp je rovnaký, pracujeme s URL adresou príslušného reportu, ktorá môže obsahovať prípadné parametre.

Jednoduchá HTML aplikácia

Tento prípad snáď netreba ani komentovať. Do kontextu HTML stránky, konkrétne do vytvoreného frame s definovanou veľkosťou vložíme URL adresu na príslušný report, v našom prípade s natvrdo zadanými parametrami

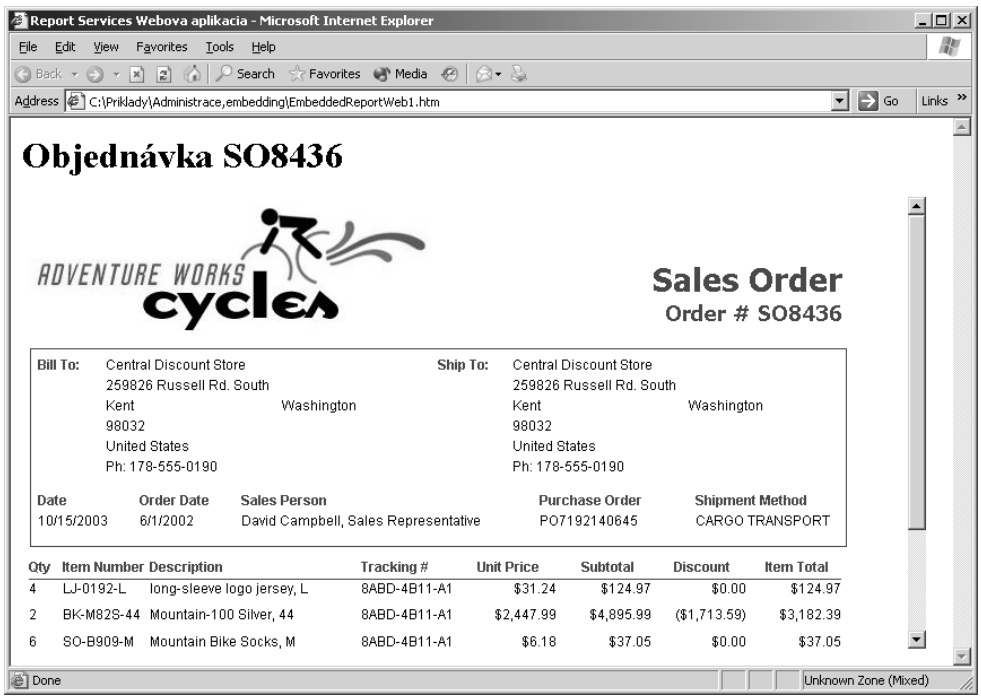
```
<html><head><title>Report Services Webova aplikacia</title>
       <meta http-equiv="Content-Type" content="text/html; charset=windows-1250">
      </head>
<body>
  <h1>Objednávka SO8436</h1>
<p><iframe name="I1" height="400" width="800"
src="http://ll2k3/ReportServer?%2fSampleReports%2fSales+Order+Detail&SalesOrderNumber=SO843
6&rc:parameters=false">
 Problem s  podporou inline frame
</iframe></p>
</body></html>
```



Obr. 5.3. Pripojenie reportu do HTML stránky

prípadne môžeme pripojiť report do kontextu webovej stránky bez toolbaru, kedy nastavíme parameter **toolbar=false**

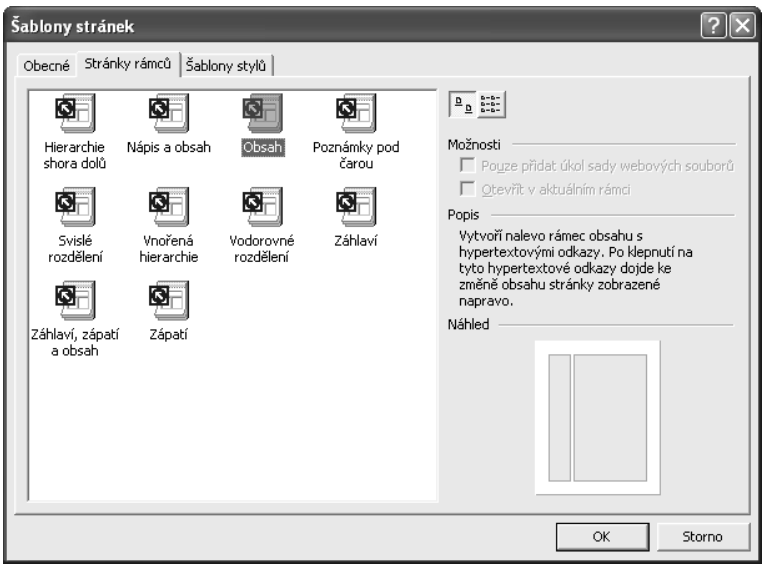
```
<html><head><title>Report Services Webova aplikacia</title>
       <meta http-equiv="Content-Type" content="text/html; charset=windows-1250">
      </head>
<body>
  <h1>Objednávka SO8436</h1>
<p><iframe name="I2" height="400" width="800"
src="http://ll2k3/ReportServer?%2fSampleReports%2fSales+Order+Detail&SalesOrderNumber=SO843
6&rc:toolbar=false">
  Problem s podporou inline frame
</iframe></p>

</body></html>
```



Obr. 5.4. Pripojenie reportu do HTML stránky bez toolbaru

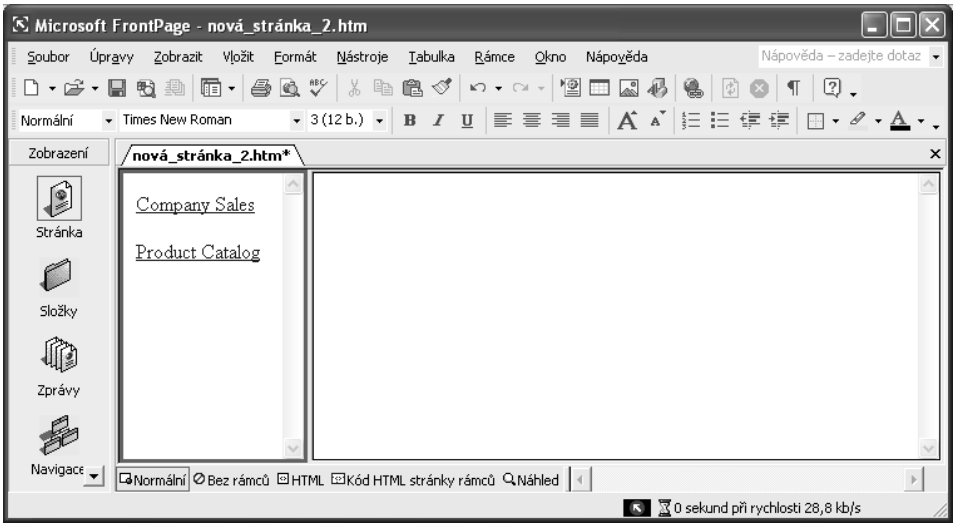
Komplexná HTML aplikácia

Ako ďalší príklad ukážeme námet na vytvorenie komplexnejšej HTML aplikácie (napríklad pomocou nástroja Microsoft Front Page), ktorá bude využívať pripojené reporty. V tomto prípade sa podstatne rozšíri komfort vývoja webovej aplikácie a aj jej technologické možnosti. V aplikácii MS Front Page vytvoríme novú aplikáciu v takom dizajne ako požadujeme. My sme využili šablóny stránok rámcov, pričom sme pre jednoduchosť vybrali také usporiadanie, kedy v ľavom rámci sú odkazy na jednotlivé reporty a v pravom rámci sa zobrazí report



Obr. 5.5. MS Font Page – výber typu ramcov zo šablóny stránok

Do takto pripravenej stránky do ľavého Frame jednoducho povkladáme odkazy na jednotlivé reporty, prípadne iné ovládacie prvky, napríklad na zadávanie parametrov a podobne.



Obr. 5.6. MS Font Page – návrh stránky s dvomi rámcami

Samozrejme tu sa možnosti návrhu len začínajú. Okrem grafického dizajnu a zobrazenia potrebných informácií mimo reportu je možné napríklad zadávať parametre, stránkovať, meniť typ reportu (tabuľka, graf...) a podobne. V našom jednoduchom prípade sme vytvorili tri stránky. Stránka **index.html** vytvára obidva rámce

```
<html><head>
<meta name="GENERATOR" content="Microsoft FrontPage 5.0">
<meta name="ProgId" content="FrontPage.Editor.Document">
<title>Nová stránka 2</title>
</head>

<frameset cols="150,*">
  <frame name="obsah" target="hlavní" src="menu.html">
  <frame name="hlavní" src="pravy.html">
  <noframes>
  <body>

  <p>Na této stránce jsou použity rámce, prohlížeè je však nepodporuje.</p>

  </body>
  </noframes>
</frameset>

</html>
```

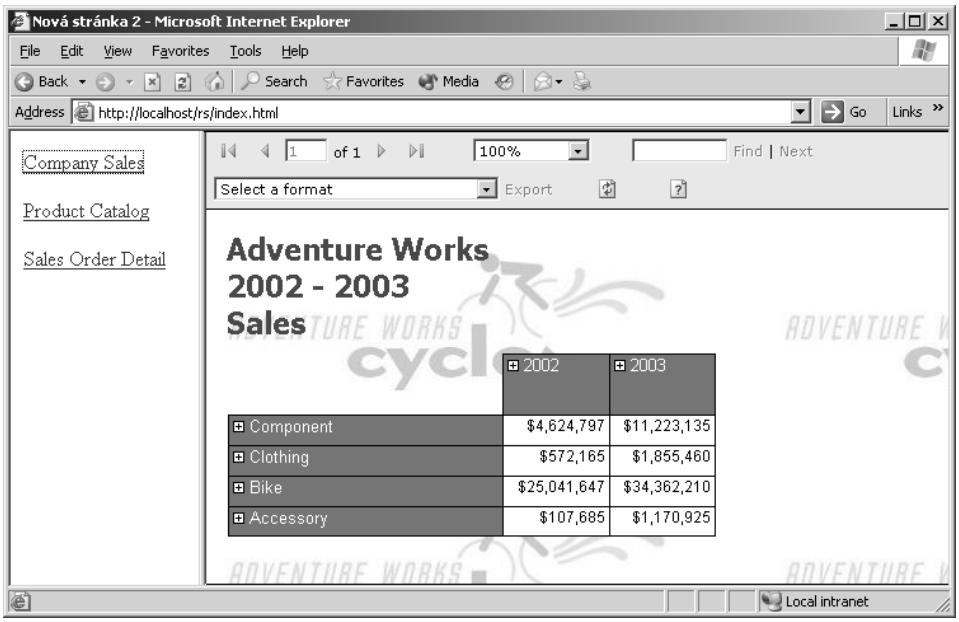
na stránke **menu.html** je obsah ľavého frame, teda zoznam odkazov na reporty

```
<html>

<head>
<meta name="GENERATOR" content="Microsoft FrontPage 5.0">
<meta name="ProgId" content="FrontPage.Editor.Document">
<meta http-equiv="Content-Type" content="text/html; charset=windows-1252">
<title>Nová stránka 4</title>
<base target="hlavní">
</head>

<body>
<p><a href="http://ll2k3/ReportServer?/SampleReports/Company+Sales">Company
Sales</a></p>
```

```
<p><a href="http://ll2k3/ReportServer?/SampleReports/Company+Sales">Product
Catalog</a></p>
<p><a href="http://ll2k3/ReportServer?%2fSampleReports%2fSales+Order+Detail">Sales Order
Detail</a></p>

</body>
</html>
```

a na stránke **pravy.html** je obsah úvodnej stránky pred zobrazením prvého reportu.
V našom príklade je tam pre jednoduchosť len nadpis

```
<HTML><BODY>
  <H1>Reporty s vyuzitim reportovacich sluzieb</FONT></H1>
</BODY></HTML>
```



Obr. 5.7. Príklad webovej aplikácie

Rovnako by sme pracovali s URL adresou v ľubovoľnom type serverovej aplikácie, napríklad v ASP.NET

ASP.NET aplikácia

V adresári

C:\Program Files\Microsoft SQL Server\MSSQL\Reporting Services\Samples\Applications\ReportViewer

nájdeme medzi ukážkovými príkladmi aj príklad aplikácie ReportViewer. Ako prvý krok otvoríme vo vývojovom prostredí Visual studio.NET 2003 súbor ReportViewer.sln a aplikáciu následne preložíme a zostavíme. Výsledkom prekladu bude DLL knižnica ReportViewer.dll, ktorá bude umiestnená v adresári

C:\Program Files\Microsoft SQL Server\MSSQL\Reporting Services\Samples\Applications\ReportViewer**cs**\bin\debugg

alebo

C:\Program Files\Microsoft SQL Server\MSSQL\Reporting Services\Samples\Applications\ReportViewer**cs**\bin\release

Prípadne ak miesto C# použijeme programovací jazyk VB, v podadresári sa CS zmení na VB

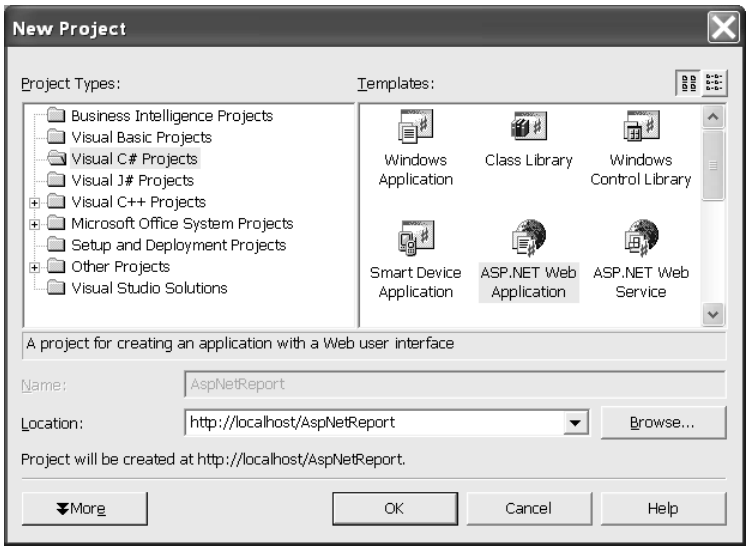
vb\bin\debugg

alebo

vb\bin\release

V podstate sa jedná o objektovo orientovaný generátor IFRAME tagov.

Použitie dll – ky ReportViewer si ukážeme na praktickom príklade. Vo vo vývojovom prostredí Visual studio.NET 2003 vytvoríme nový projekt typu ASP.NET Web Application s názvom napríklad AspNetReport.

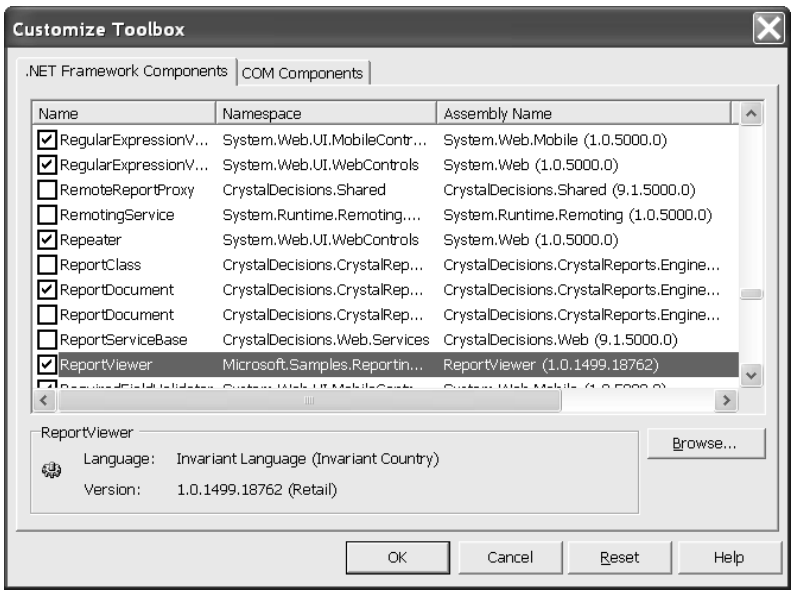


Obr. 5.8. Vytvorenie novej ASP.NET webovej aplikácie

Aplikácia bude potom prístupná na URL adrese http://localhost/AspNetReport

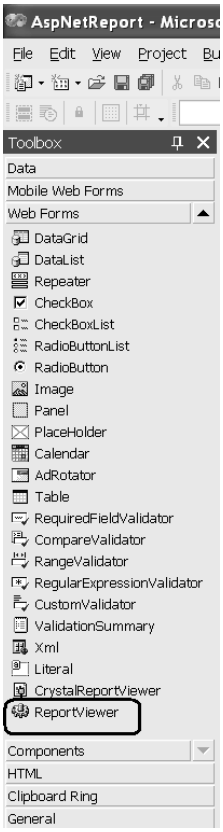
A súbory, ktoré túto aplikáciu tvoria budú uložené v adresári C:\Inetpub\wwwroot\AspNetReport

Po vytvorení novej aplikácie pridáme do projektu referenciu na Report Viewer pomocou položky **Add/Remove Toolbox Items** v menu **Tools**.



Obr. 5.9. Pridanie referencie na Report Viewer

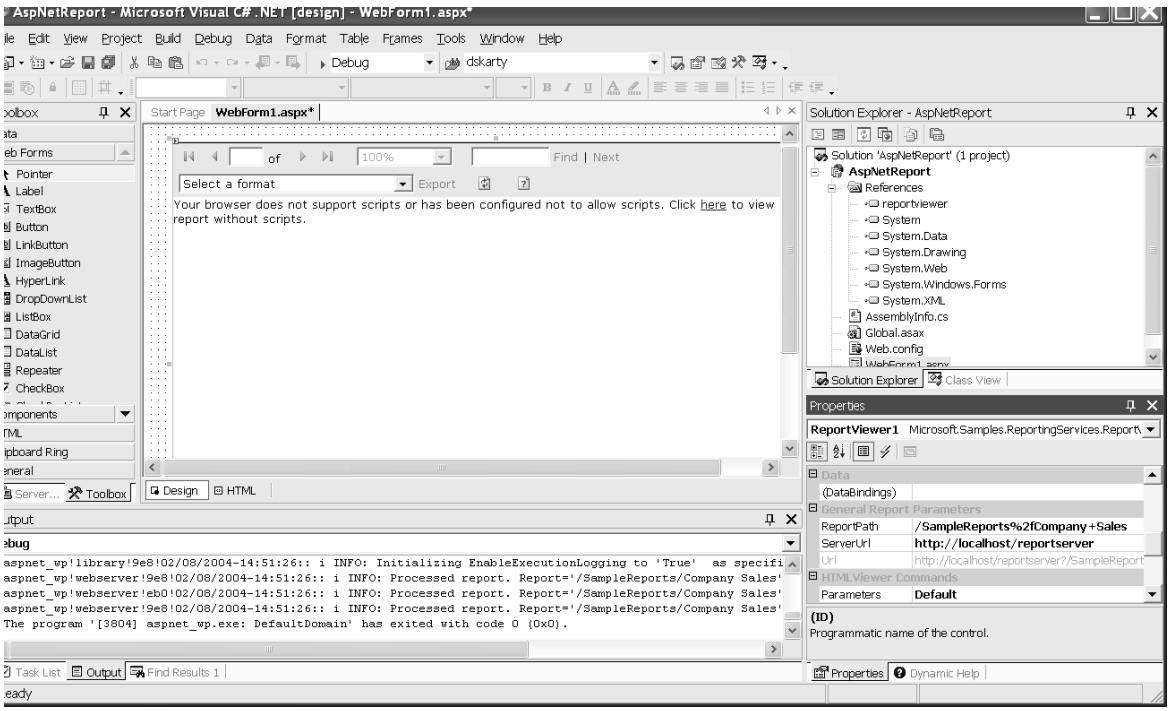
Obr. 5.10. Pridanie referencie na Report Viewer



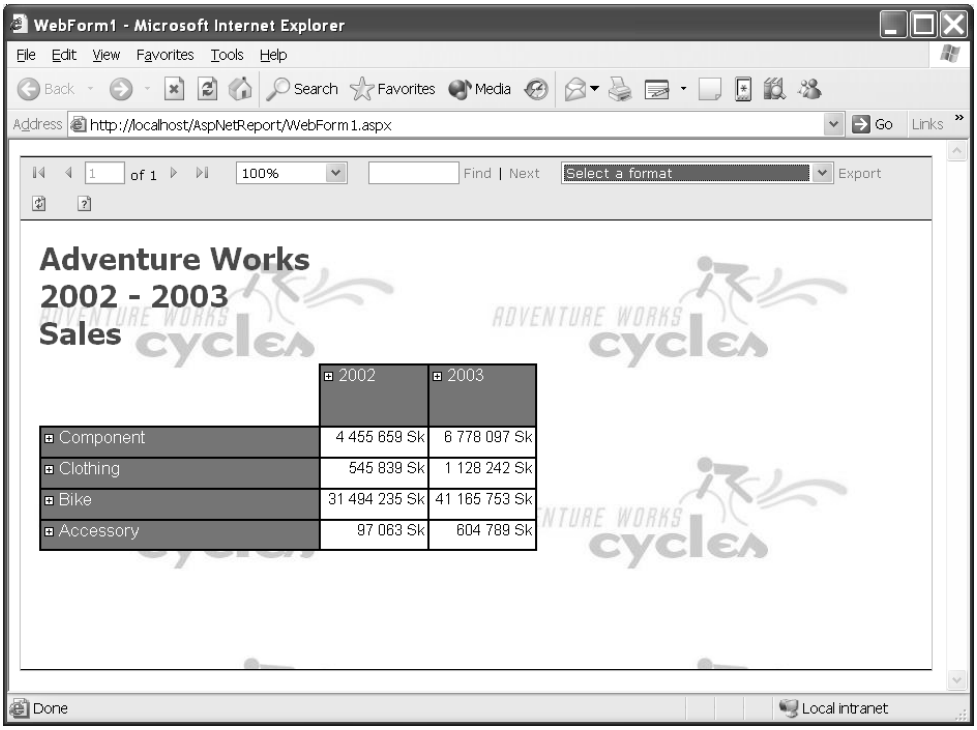
Po týchto úkonoch v okne Toolboxu pribudne nevizuálna komponenta ReportViewer.

Ak túto nevizuálnu komponentu umiestnime na plochu ASP.NET aplikácie zobrazí sa oznam, aby sme nastavili Server URL a Report Path

Server URL: http://localhost/reportserver
Report Path: /SampleReports/CompanySales



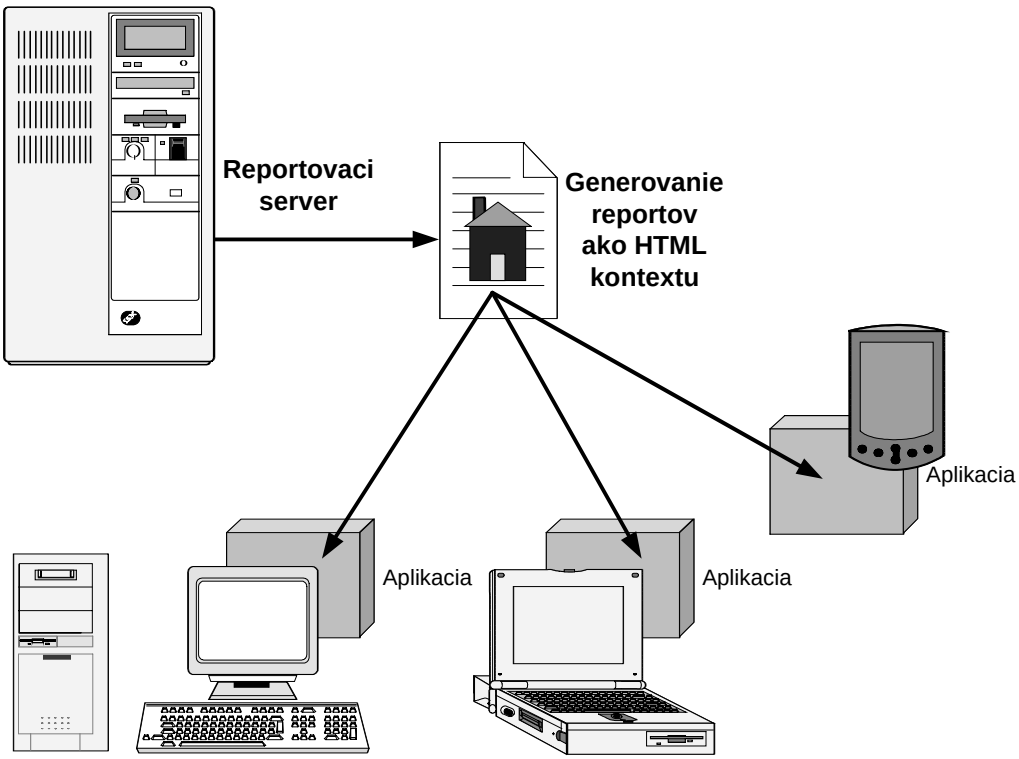
Obr. 5.11. Návrh ASP.NET aplikácie



Obr. 5.12 Test ASP.NET aplikácie

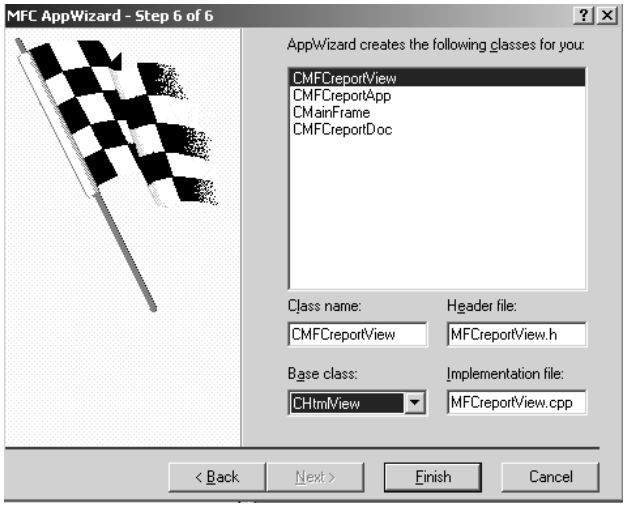
Po nastavení parametrov môžema ASP.NET aplikáciu otestovať

Embedding do aplikácií a riešení typu „tlustý klient“



Obr. 5.13. Príklad architektúry pre využívanie reportovacích služieb v aplikácii typu „tlustý klient“

Jadro problému zobrazenia reportov je u aplikácií typu tlustý klient zobrazenie kontextu prehliadača HTML stránok v okne aplikácií. Táto špecifikácia nám deklaruje úplnú platformovú nezávislosť, čo sa týka vývoja aplikácií. Aby sme túto nezávislosť aj prakticky demonštrovali, náš prvý príklad bude tak trochu historický. Ukážeme pripojenie aplikácie vyvinutej vo vývojovom prostredí Visual C++ k reportovaciemu serveru. Navrhneme aplikáciu využívajúcu objektovú knižnicu MFC (Microsoft Foundation Class). V sprievodcovi návrhom aplikácie navrhneme MFC aplikáciu využívajúcu Document – View architektúru. Jedinou zmenou oproti implicitným krokom, ktoré nám ponúka sprievodca je zmena triedy CView pre zobrazovanie obsahu dokumentu na odvodenú triedu CHtmlView. Táto trieda je určená pre zobrazenie webového kontextu v okne MFC aplikácie



Obr. 5.14. Príklad návrhu MFC aplikácie vo vývojovom prostredí Visual C++ 6.0

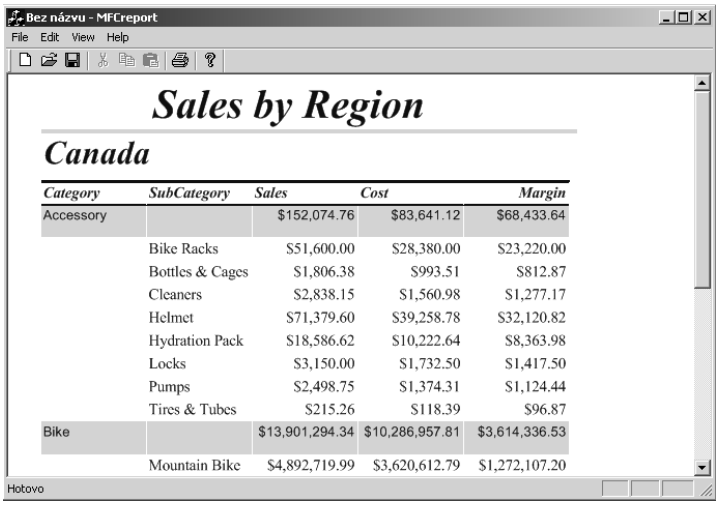
Prvou úpravou sprievodcom navrhnutej aplikácie bude v triede CHtmlView

```
void CMFCreportView::OnInitialUpdate()
{
	CHtmlView::OnInitialUpdate();

	// TODO: This code navigates to a popular spot on the web.
	//  change the code to go where you'd like.
	Navigate2(_T("http://www.microsoft.com/visualc/"),NULL,NULL);
}
```

V metóde **OnInitialUpdate**()nahradíme preddefinovanú URL adresu URL adresou príslušného reportu, napríklad

```
void CMFCreportView::OnInitialUpdate()
{
	CHtmlView::OnInitialUpdate();
	Navigate2(_T("http://localhost/reportserver/SampleReports/Company
Sales?rs:Command=Render"),NULL,NULL);
}
```

Týmto úkonom je prvý krok v návrhu MFC aplikácie využívajúcej reporty ukončený. Či sme boli úspešní, môžeme jednoducho vyskúšať spustením aplikácie.



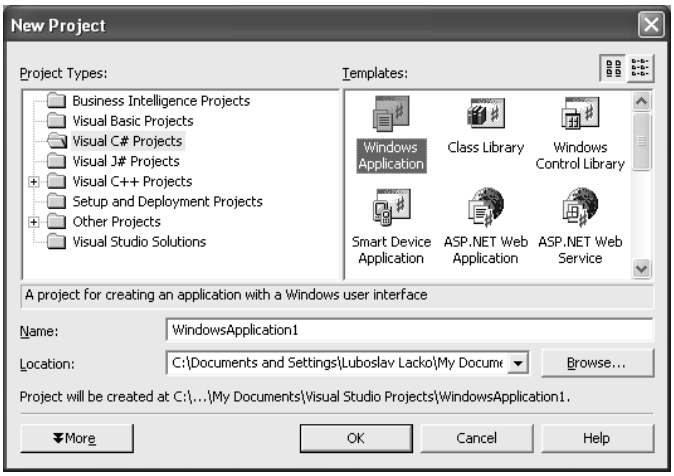
Category	SubCategory	Sales	Cost	Margin
Accessory		$152,074.76	$83,641.12	$68,433.64
	Bike Racks	$51,600.00	$28,380.00	$23,220.00
	Bottles & Cages	$1,806.38	$993.51	$812.87
	Cleaners	$2,838.15	$1,560.98	$1,277.17
	Helmet	$71,379.60	$39,258.78	$32,120.82
	Hydration Pack	$18,586.62	$10,222.64	$8,363.98
	Locks	$3,150.00	$1,732.50	$1,417.50
	Pumps	$2,498.75	$1,374.31	$1,124.44
	Tires & Tubes	$215.26	$118.39	$96.87
Bike		$13,901,294.34	$10,286,957.81	$3,614,336.53
	Mountain Bike	$4,892,719.99	$3,620,612.79	$1,272,107.20

Obr. 5.15. Príklad najjednoduchšej MFC aplikácie využívajúcej reportovacie služby

ak sa report zobrazí správne, môžeme pokračovať v návrhu ďalšej aplikačnej logiky a navrhnúť napríklad ovládacie prvky pre zobrazenie niektorého konkrétneho reportu z palety reportov, ktoré sú k dispozícii prostredníctvom reportovacieho servera. Aplikačné rozhranie taktiež poskytuje komfortné možnosti pre zadávanie parametrov a podobne. Po zadaní parametrov, ich prípadnej verifikácii a potvrdení nám jadro aplikačnej logiky skonštruuje nový reťazec URL adresy a v okne aplikácie sa následne zobrazí aktualizovaný report

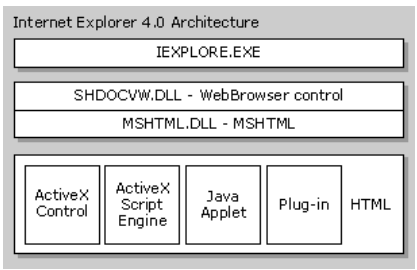
Návrh .NET aplikácie typu tlustý klient

Pri návrhu aplikačného formulára v programovacom jazyku Visual Basic.NET, alebo Visual C# potrebujeme na jeho plochu pridať komponentu pre prehliadanie webového obsahu. Prvý krok pre vytvorenie nového projektu je štandardný.



Obr. 5.16. Vytvorenie aplikácie typu hrubý klient v programovacom jazyku Visual C#

Formulár a aplikačnú logiku navrhneme podľa potrieb aplikácie. Nami popisovaná problematika sa zúži na zobrazenie okna webového prehliadača vo formulári aplikácie. Princíp popisuje schéma



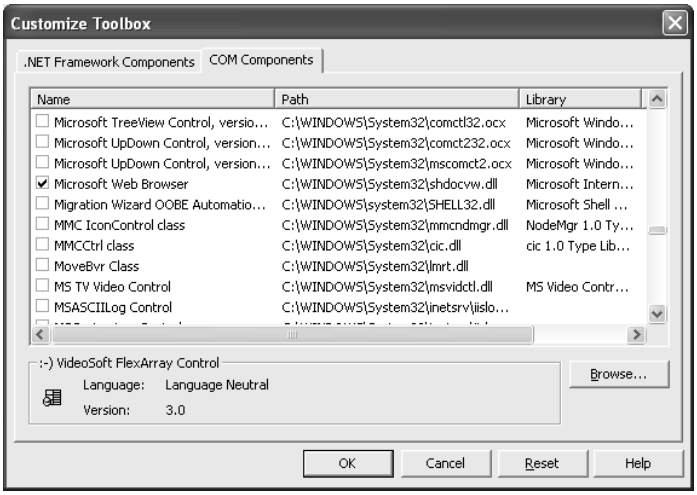
Obr. 5.17. Architektúra komponenty prehliadača webového obsahu

Podrobnejšie informácie sa dajú nájsť v MSDN napríklad na adresách

http://msdn.microsoft.com/workshop/browser/webbrowser/RefList_VB.asp

http://msdn.microsoft.com/library/default.asp?url=/workshop/browser/webbrowser/RefList_VB.asp

V okne Solution Explorer klikneme pravým tlačidlom myši na zložku References folder a aktivujeme voľbu Add Reference. Pridáme komponentu **Microsoft.mshtml**. Následne do Toolboxu pridáme komponentu **Microsoft Web Browser**



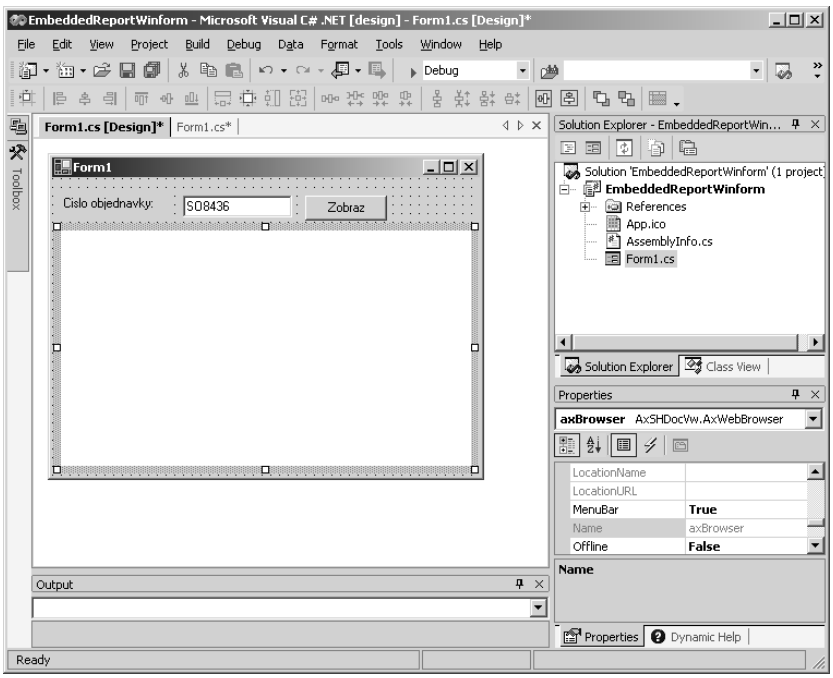
Obr. 5.18. Pridanie komponenty Microsoft Web Browser do Toolboxu

Nakoniec komponentu vhodne umiestnime na plochu aplikačného formulára.

V inicializačnej časti formulára bude kód

```
public class Form1 : System.Windows.Forms.Form
{
      private AxSHDocVw.AxWebBrowser axBrowser;
      ...
      ...
```

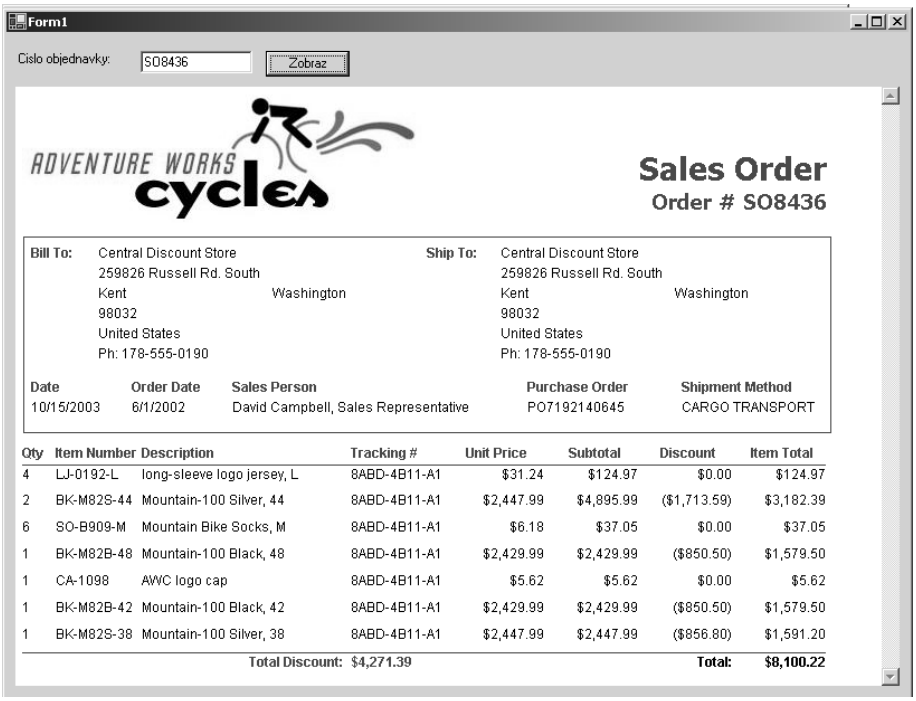
V našom príklade sme na plochu formulára pridali editačné okno pre zadávanie čísla objednávky a tlačidlo pre jeho potvrdenie



Obr. 5.19. Návrh formulára tlustého klienta pre prehliadanie reportov objednávok

Celé jadro aplikačnej logiky potom spočíva v obsluhe udalosti kliknutia na toto tlačidlo

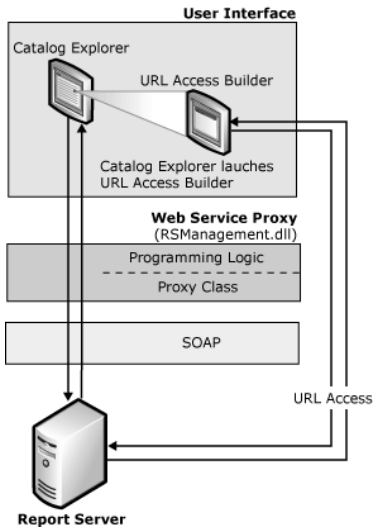
```
private void button1_Click(object sender, System.EventArgs e)
{
       Object optional = System.Reflection.Missing.Value;
       string
url=@"http://ll2k3/ReportServer?%2fSampleReports%2fSales+Order+Detail&SalesOrderNumber="+tx
tCisloObjednavky.Text+@"&rc:toolbar=false";
     axBrowser.Navigate(url,ref optional,ref optional,ref optional,ref optional);
}
```



Obr. 5.20. Aplikácia typu hrubý klient pre prehliadanie reportov objednávok

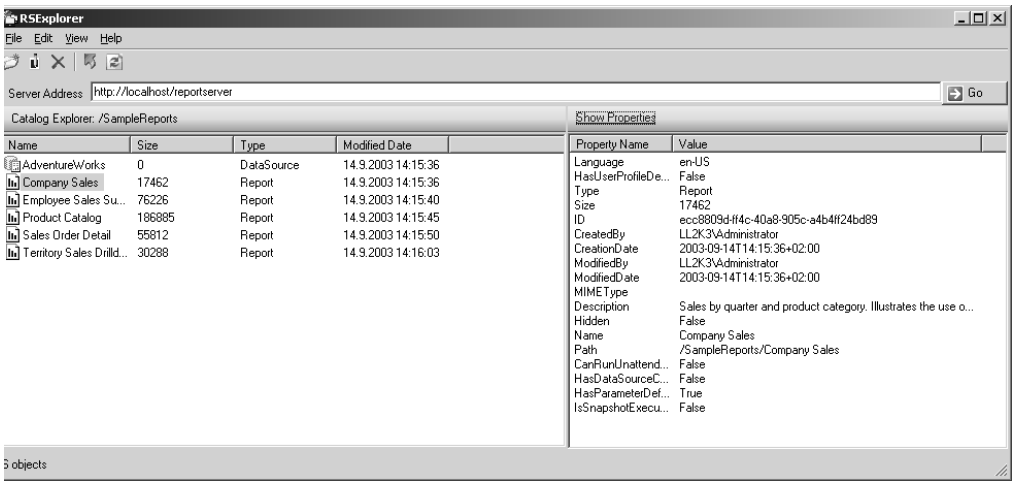
Komplexná viacvrstvová aplikácie

Spolu s reportovacími službami je dodávaný aj príklad (RSExplorer) aplikácie s viacvrstvovou architektúrou. Účelom aplikácie je prehliadať si jednotlivé reporty pod správou reportovacieho servera a vypísať, prípadne zmeniť niektoré ich vlastnosti. Používateľ prichádza do styku s vrstvou používateľského rozhrania, kedy pracuje s reportami. Táto vrstva v podstate len zobrazuje informácie potrebné k výberu vhodného reportu, presnejšie k sformulovaniu URL adresy s reportom.



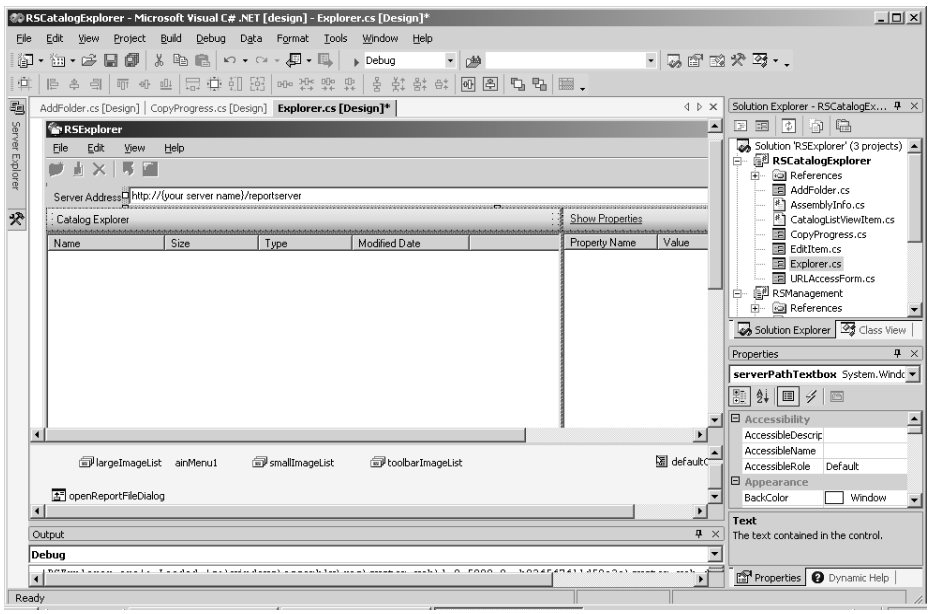
Obr. 5.21. Príklad trojjvrstvovej architektúry aplikácie využívajúcej webové služby

Aplikácia svojim dizajnom a princípom činnosti pripomína kombináciu prehliadača webového obsahu a súborového manažéra. V hornej časti je pole pre zadanie URL adresy reportovacieho servera. V ľavom listboxe je zoznam reportov a v pravom listboxe môžeme vypísať vlastnosti a parametre pre vybraný report.



Obr. 5.22. Príklad RS Explorer

Celú aplikáciu okrem pomocných dialógov, napríklad na zmenu popisu reportu tvorí jeden aplikačný formulár



Obr. 5.23. Návrh aplikačného formulára príkladu RS Explorer vo vývojovom prostredí Visual Studio .NET 2003

Po zadaní URL adresy reportovacieho servera a jej potvrdení tlačidlom GO sa aktivuje procedúra

```
private void goButton_Click(object sender, System.EventArgs e)
{
      this.ConnectToServer();
}

private void ConnectToServer()
{
      //Init UI
      EnableButtons(true);
      this.Path = Root;
      string serverPath = serverPathTextbox.Text;
      try
```

```
        {
                //Connect to RS
                Cursor.Current = Cursors.WaitCursor;
                this._catalogManager =  new CatalogManager();
                this._catalogManager.Credentials = CredentialCache.DefaultCredentials;
                //See CatalogManager.Connect() for details on connecting to the RSWebService
                //A production application would perform a complete check on the url path
                this._catalogManager.Connect(serverPath);
                //Display root items
                DisplayItems( Path );
        }
        catch(Exception ex)
        {
                EnableButtons(false);
                this.HandleGeneralException(ex);
        }
        finally
        {
                Cursor.Current = Cursors.Default;
        }

}
```

Ak si chceme niektorý s reportov zobraziť, klikneme na jeho názov a v samostatnom formulári sa vyrendruje report príkazom Render. URL adresa napríklad pre report Company Sales potom bude

http://localhost/reportserver/SampleReports/Company Sales?rs:Command=Render

Rozšíriteľnosť Reportovacích služieb

Skutočnosť, že Reportovacie služby sú založené na platforme .NET Framework a disponujú otvorenou architektúrou, im dáva veľké možnosti flexibility a škálovateľnosti. Možnosti rozšírenia sú hlavne v týchto oblastiach
– bezpečnosť
– data provider
– rendering
– doručovanie reportov

Ukážkové príklady na rozšíriteľnosť reportovacích služieb sú v adresári

C:\Program Files\Microsoft SQL Server\MSSQL\Reporting Services\Samples\Extensions

Postup je vysvetlený v dokumentácii na adrese

ms-help://MS.RSBOL80.1033/RSAMPLES/htm/rss_sampleapps_v1_16g2.htm

Materiály na webe

Zdroj	Adresa
Microsoft Support Services	http://support.microsoft.com/directory
Microsoft newsgroups	news://news.microsoft.com/
Microsoft Windows Hardware Compatibility List	http://www.microsoft.com/hcl
Microsoft MSDN®	http://msdn.microsoft.com
Professional Association for SQL Server	http://www.sqlpass.org/
Microsoft SQL Server Developer Center	http://msdn.microsoft.com/sql/
SQL Server Magazine	http://www.sqlmag.com/
Microsoft SQL Server Support	http://support.microsoft.com/support/sql
TechNet site	http://www.microsoft.com/technet/
Microsoft SQL Server	http://www.microsoft.com/sql

Prílohy

Záver

Doplnenie MS SQL Servera 2000 o Reportovacie služby významnou mierou prispeje k dostupnosti údajov pre široké spektrum ich používateľov. Hlavnou výhodou je však skutočnosť, že reportovacie služby budú implementované do novej verzie SQL serveru s kódovým označením „YUKON“, takže zvádnutím tejto technológie sa de facto pripravíme na efektívnu migráciu v oblasti generovania reportov na základe údajov z relačných a analytických databáz.

Prílohy

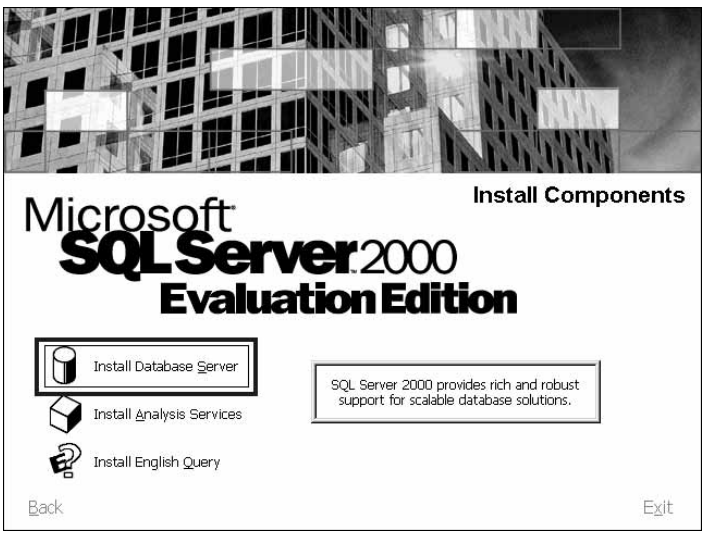
Príloha CD:

Inštalácia 120 dňovej verzie Microsoft SQL Serveru 2000

Inštalácia pre operačný systém Windows 2000, alebo Windows XP Professional je pomerne bezproblémová, stačí postupovať podľa pokynov inštalačného programu a vo väčšine prípadov akceptovať implicitné nastavenia.

Požiadavky na hardvérové vybavenie počítača

Procesor: Intel kompatibilný 166 MHz a vyšší
Pamäť RAM: 128 MB a viac.
Operačný systém: Microsoft Windows NT Workstation 4.0 alebo Windows NT Server 4.0, alebo Windows NT Server 4.0 Enterprise Edition, všetko s nainštalovaným rozšírením (servis packom) SP5 alebo vyšším, alebo Microsoft Windows 2000 Professional, Server, Windows 2000 Advanced Server prípadne Windows 2000 Datacenter Server. (Trial verzia totiž na rozdiel od komerčnej Enterprise verzie funguje taktiež na počítačoch s Windows NT Workstation a Windows 2000 Professional).

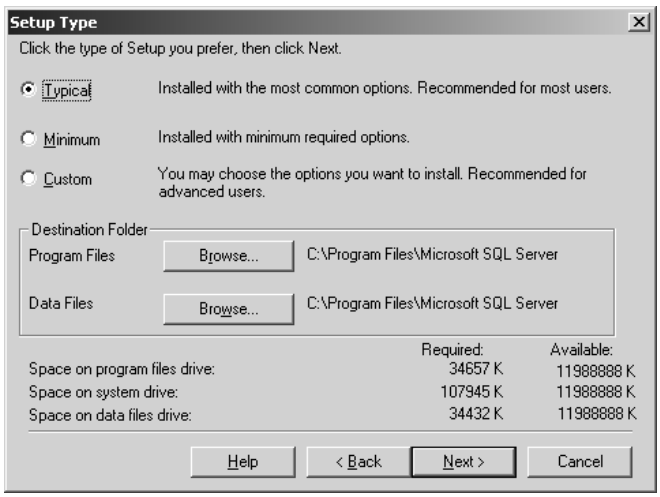


Obr. P.1 – Inštalácia SQL Servera

Ak nemáme k dispozícii firemný server, zvolíme si inštaláciu na lokálny počítač, na ktorom budeme pracovať s databázou. Vyberieme najjednoduchší spôsob inštalácie: ***Create a new instance of SQL Server, or install Client Tools*** *(Vytvor novú inštanciu SQL Serveru, alebo nainštaluj klientské nástroje)*. Po súhlase s licenčným ujednaním nasleduje ďalší dialóg v ktorom zvolíme preddefinovanú možnosť ***Server And Clients Tools*** *(Server a klientské nástroje)*. Ak nie sme skúsenými administrátormi, v ďalšom dialógovom okne si vyberieme typickú inštaláciu. Zároveň sa dozvieme približné nároky SQL Servera na diskovú kapacitu, ktorá je vzhľadom na dostupné kapacity dnešných komerčných diskov. V našom prípade sa jednalo o položky:

Program files drive	34 MB
System drive	107 MB
Data files drive	34 MB

čiže spolu niečo pod 200 MB.

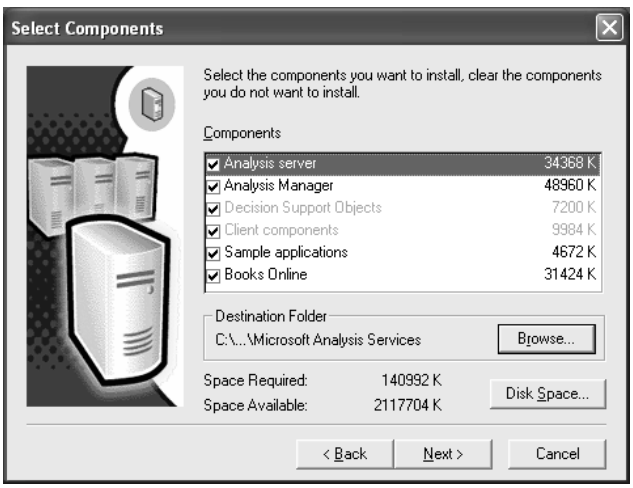


Obr. P.2 – Typy inštalácií SQL Servera

V okne Services Accounts zvolíme možnosť ***Domain User Account***, zadáme prihlasovacie heslo do systému Windows 2000. Autentifikáciu nastavíme najlepšie na Windows NT mode, v prípade špeciálnych požiadaviek na ***Mixed mode***, kedy sa overí autentifikácia operačného systému Windows NT, alebo Windows 2000 spolu s prihlasovacím heslom SQL Serveru. Zadáme heslo, ktoré budeme používať. Tento údaj je zároveň posledný, ktorý musíme pri inštalácii MS SQL Serveru 2000 zadať.

Inštalácia analytických služieb

Hoci táto publikácia je zameraná predovšetkým na reportovacie služby, môžeme z inštalačného CD databázového servera SQL Server 2000 nainštalovať aj analytické služby. Znovu spustíme inštalačný server ale tentokrát vyberieme možnosť „Install Analysis Services“. Inštalátor vykoná všetko za nás. V dialógu Select Components prípadne môžeme zmeniť parametre, ale implicitne sa nainštaluje všetko potrebné



Obr. P.3 – Inštalácia komponentov analytických služieb

Inštalácia servisného balíčka Service Pack 3

Ako vyplýva z čísla typového označenia SQL Servera 2000 jedná sa o produkt osvedčený takmer trojročnou prevádzkou. Za tento čas pripravila firma Microsoft množstvo rôznych záplat a vylepšení, preto je potrebné nainštalovať z inštalačného CD aj Service Pack 3. Tento balíček sa montuje osobitne pre SQL Server a osobitne pre analytické služby.

Štruktúra cvičnej databázy Northwind

V niektorých príkladoch v tejto publikácii budeme vytvárať reporty aj nad údajmi z cvičnej databázy fiktívnej firmy **Northwind**. V tejto databáze budeme pracovať s tabuľkami **Employees, Orders** a **Order Details**. Tabuľka **Employees** obsahuje údaje o zamestnancoch firmy. Jej návrhová štruktúra je nasledovná:

```
EmployeeID          int
LastName            nvarchar (20)
FirstName           nvarchar (10)
Title               nvarchar (30)
TitleOfCourtesy     nvarchar
BirthDate           datetime
HireDate            datetime
Address             nvarchar (60)
City                nvarchar (15)
Region              nvarchar (15)
PostalCode          nvarchar (10)
Country             nvarchar (15)
HomePhone           nvarchar (24)
Extension           nvarchar (4)
Photo               image
Notes               ntext
ReportsTo           int
PhotoPath           nvarchar (255)
```

Tabuľka obsahuje 9 záznamov a pretože niektoré z nich sú pomerne rozsiahle (napríklad stĺpec Notes) a foto sa v textovej podobe ani jednoducho vyjadriť nedá vypíšeme len obsah niekoľkých stĺpcov, napríklad príkazom

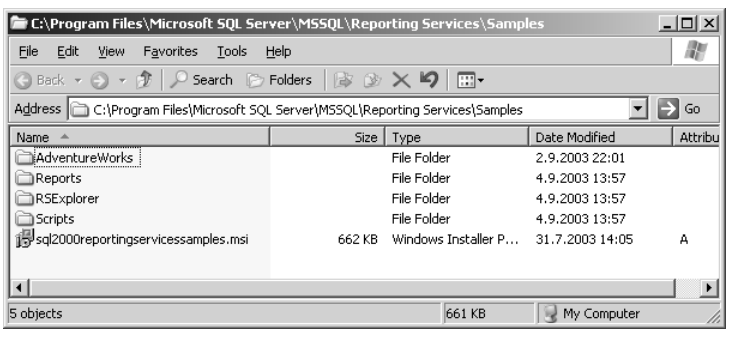
```
SELECT LastName, FirstName, Title, Address, City, Country
FROM Employees
```

```
LastName   FirstName  Title                  Address                        City      Country
---------- ---------- ---------------------- ------------------------------ --------- -------
Davolio    Nancy      Sales Representative   507 - 20th Ave. E. Apt. 2A     Seattle   USA
Fuller     Andrew     Vice President, Sales  908 W. Capital Way             Tacoma    USA
Leverling  Janet      Sales Representative   722 Moss Bay Blvd.             Kirkland  USA
Peacock    Margaret   Sales Representative   4110 Old Redmond Rd.           Redmond   USA
Buchanan   Steven     Sales Manager          14 Garrett Hill                London    UK
Suyama     Michael    Sales Representative   Coventry House Miner Rd.       London    UK
King       Robert     Sales Representative   Edgeham Hollow Winchester Way  London    UK
Callahan   Laura      Inside Sales Coordinator 4726 - 11th Ave. N.E.        Seattle   USA
Dodsworth  Anne       Sales Representative   7 Houndstooth Rd.              London    UK

(9 row(s) affected)
```

Cvičné príklady

Cvičné príklady sú po implicitnej inštalácii nainštalované v adresári

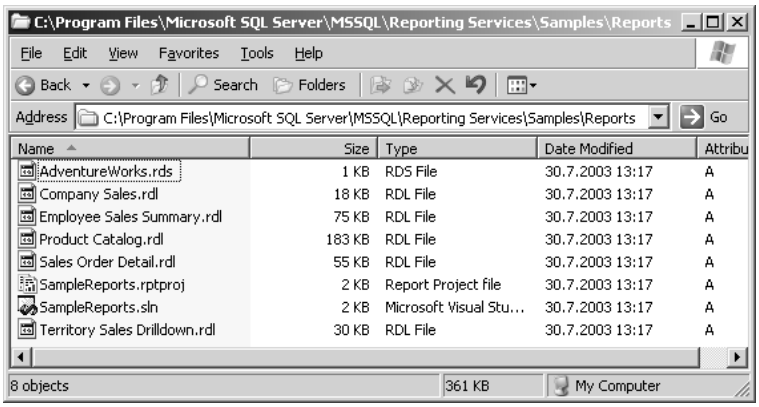
```
C:\Program Files\Microsoft SQL Server\MSSQL\Reporting Services\Samples
```



Obr. P.4 Adresár s cvičnými príkladmi

V adresári Reports sú príklady reportov.

Company Sales
Employee Sales Sumary
Product Catalog
Sales Order Detail
Territory Sales



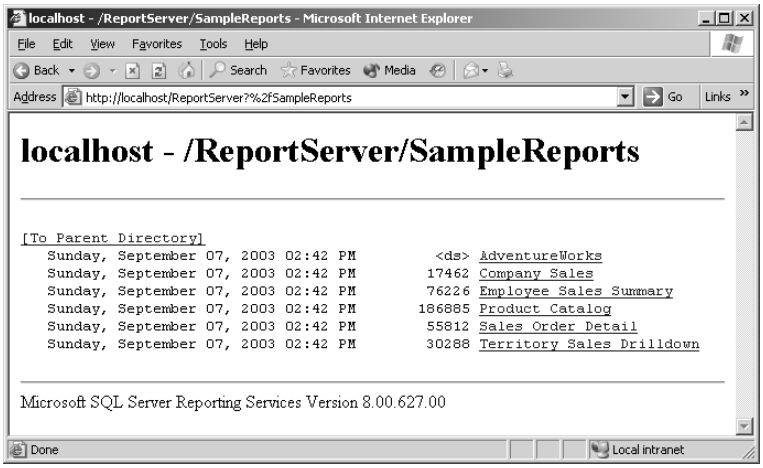
Obr. P.5 Podadresár Reports s cvičnými reportami.

Odporúčame si tieto príklady prezrieť, zostaviť a sprístupniť ich pod správu report servera. Príklady v adresári Reports sú súčasťou jedného projektu (riešenia), takže ich môžeme zaviesť do vývojového prostredia Visual Studio .NET 2003 naraz. Postup je jednoduchý. Stačí v prieskumníku súborov kliknúť na súbor

```
SampleReports.sln
```

a celý projekt sa automaticky zavedie do vývojového prostredia.

```
------ Build started: Project: SampleReports, Configuration: Debug ------

Build complete -- 0 errors, 0 warnings

------ Deploy started: Project: SampleReports, Configuration: Debug ------

Deploying to http://localhost/ReportServer?%2fSampleReports
Deploying data source '/SampleReports/AdventureWorks'
Deploying report 'Company Sales'
Deploying report 'Employee Sales Summary'
Deploying report 'Product Catalog'
Deploying report 'Sales Order Detail'
Deploying report 'Territory Sales Drilldown'
Deploy complete -- 0 errors, 0 warnings

---------------------- Done ----------------------

    Build: 1 succeeded, 0 failed, 0 skipped
    Deploy: 1 succeeded, 0 failed, 0 skipped
```



Obr. P.6 Príklady pod správou reportovacieho servera

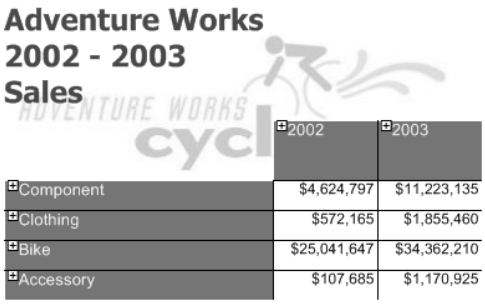
Jednotlivé príklady si stručne priblížime v našom obrázkovom prehľade.

Company Sales

Návrhové zobrazenie reportu predaja je

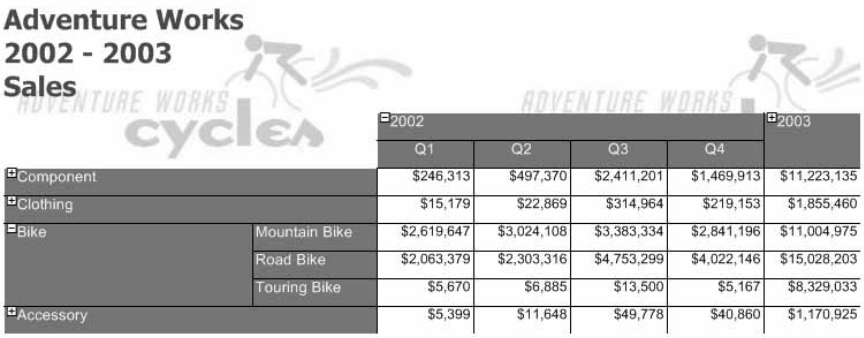
Obr. P.7 Príklad Company Sales – návrhové zobrazenie reportu

Na začiatku prehliadania máme k dispozícii len agregované sumárne výsledky pre predaj jednotlivých komodít a to pre roky 2002 a 2003



Obr. P.8 Príklad Company Sales – preview reportu (sumárny report)

Kliknutím na ikonku (+) sa môžeme vnoriť o úroveň hlbšie a zobraziť podrobnosti. Ak nás napríklad zaujíma predaj bicyklov v jednotlivých kvartáloch roku 2002, rozvinieme položku „Bike“ a rok 2002

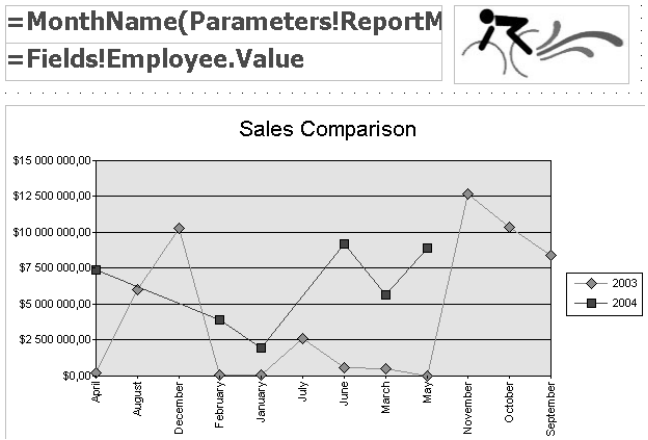


Obr. P.9 Príklad Company Sales – preview reportu (drill down riadku Bike a stĺpca roku 2002)

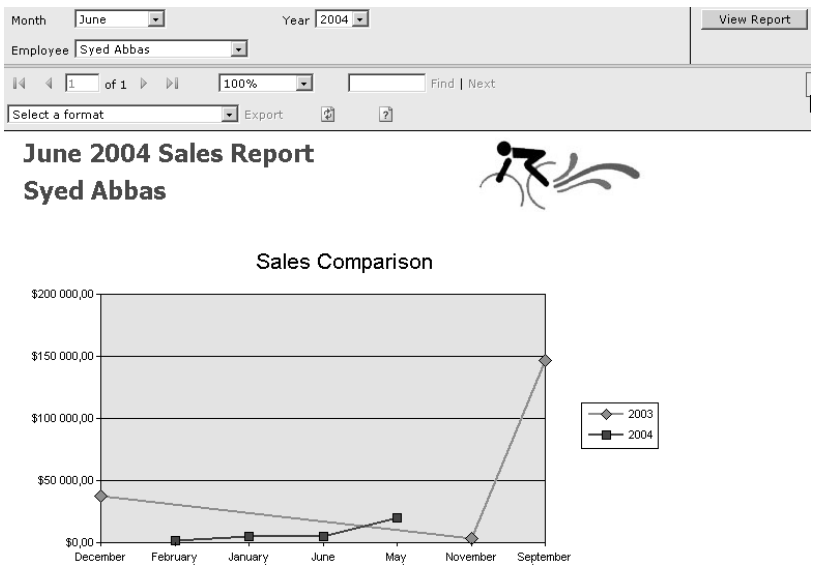
SQL dopyt pre výber údajov v tomto príklade je

```
SELECT    ProductCategory.Name AS ProdCat, ProductSubCategory.Name AS SubCat,
          DATEPART(yy, SalesOrderHeader.OrderDate) AS OrderYear,
          'Q' + DATENAME(qq, SalesOrderHeader.OrderDate) AS OrderQtr,
          SUM(SalesOrderDetail.UnitPrice * SalesOrderDetail.OrderQty) AS Sales

FROM      ProductSubCategory INNER JOIN
          SalesOrderHeader INNER JOIN
          SalesOrderDetail ON SalesOrderHeader.SalesOrderID = SalesOrderDetail.SalesOrderID
          INNER JOIN
          Product ON SalesOrderDetail.ProductID = Product.ProductID ON
          ProductSubCategory.ProductSubCategoryID = Product.ProductSubCategoryID INNER JOIN
          ProductCategory ON ProductSubCategory.ProductCategoryID =
          ProductCategory.ProductCategoryID

WHERE     (SalesOrderHeader.OrderDate BETWEEN '1/1/02' AND '12/31/03')
GROUP BY  DATEPART(yy, SalesOrderHeader.OrderDate), ProductCategory.Name,
          ProductSubCategory.Name,
          'Q' + DATENAME(qq, SalesOrderHeader.OrderDate),
          ProductSubCategory.ProductSubCategoryID
```

Employee Sales Sumary



Obr. P.10 Príklad Employee Sales Sumary – návrh reportu



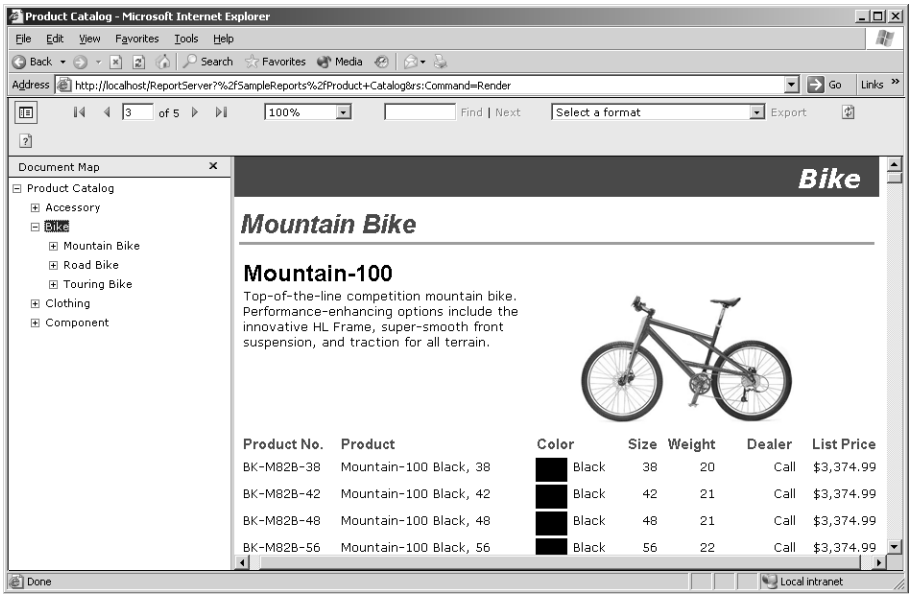
Obr. P.11 Príklad Employee Sales Sumary – Preview reportu

SQL dopyt pre výber údajov v tomto príklade je

```
SELECT      dbo.Employee.FirstName + ' ' + dbo.Employee.LastName AS Employee,
            DATEPART(Year, dbo.SalesOrderHeader.OrderDate) AS OrderYear,
            DATEPART(Month, dbo.SalesOrderHeader.OrderDate) AS OrderMonthNum,
            DATENAME(Month, dbo.SalesOrderHeader.OrderDate) AS OrderMonth,
            SUM(dbo.SalesOrderDetail.LineTotal) AS Sales

FROM  dbo.SalesOrderHeader INNER JOIN
     dbo.SalesOrderDetail ON dbo.SalesOrderHeader.SalesOrderID = dbo.SalesOrderDetail.SalesOrderID
          INNER JOIN
    dbo.SalesPerson ON dbo.SalesOrderHeader.SalesPersonID = dbo.SalesPerson.SalesPersonID INNER JOIN
    dbo.Employee ON dbo.SalesPerson.SalesPersonID = dbo.Employee.EmployeeID

WHERE       (DATEPART(Year, dbo.SalesOrderHeader.OrderDate) = @ReportYear - 1 OR
             DATEPART(Year, dbo.SalesOrderHeader.OrderDate) = @ReportYear AND
             DATEPART(Month, dbo.SalesOrderHeader.OrderDate) <= @ReportMonth) AND
            (dbo.SalesOrderHeader.SalesPersonID = @EmpID)

GROUP BY    dbo.Employee.FirstName + ' ' + dbo.Employee.LastName,
            dbo.SalesOrderHeader.SalesPersonID, DATEPART(Year,
            dbo.SalesOrderHeader.OrderDate), DATEPART(Month, dbo.SalesOrderHeader.OrderDate),
            DATENAME(Month, dbo.SalesOrderHeader.OrderDate)
```

Product Catalog

Obr. P.12 Príklad Product Catalog – návrh reportu (prvá strana)

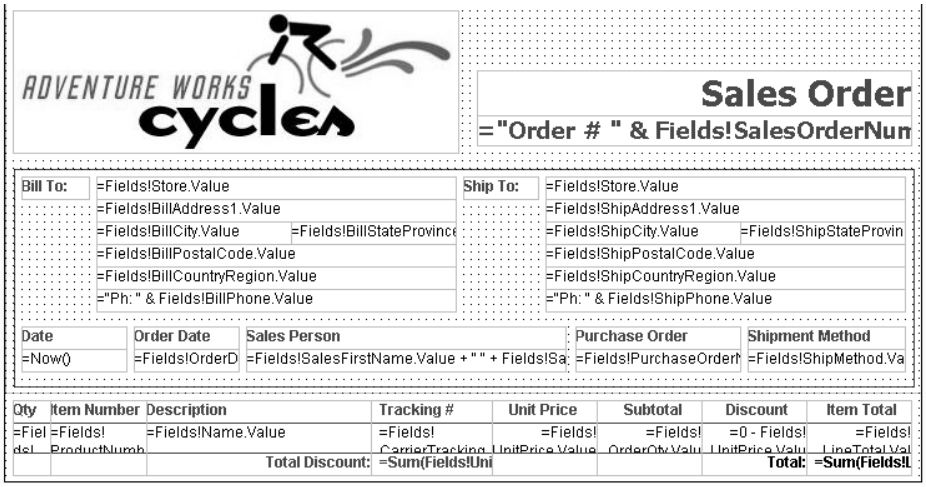


Obr. P.13 Príklad Product Catalog – preview reportu (ponuka horských bicyklov)

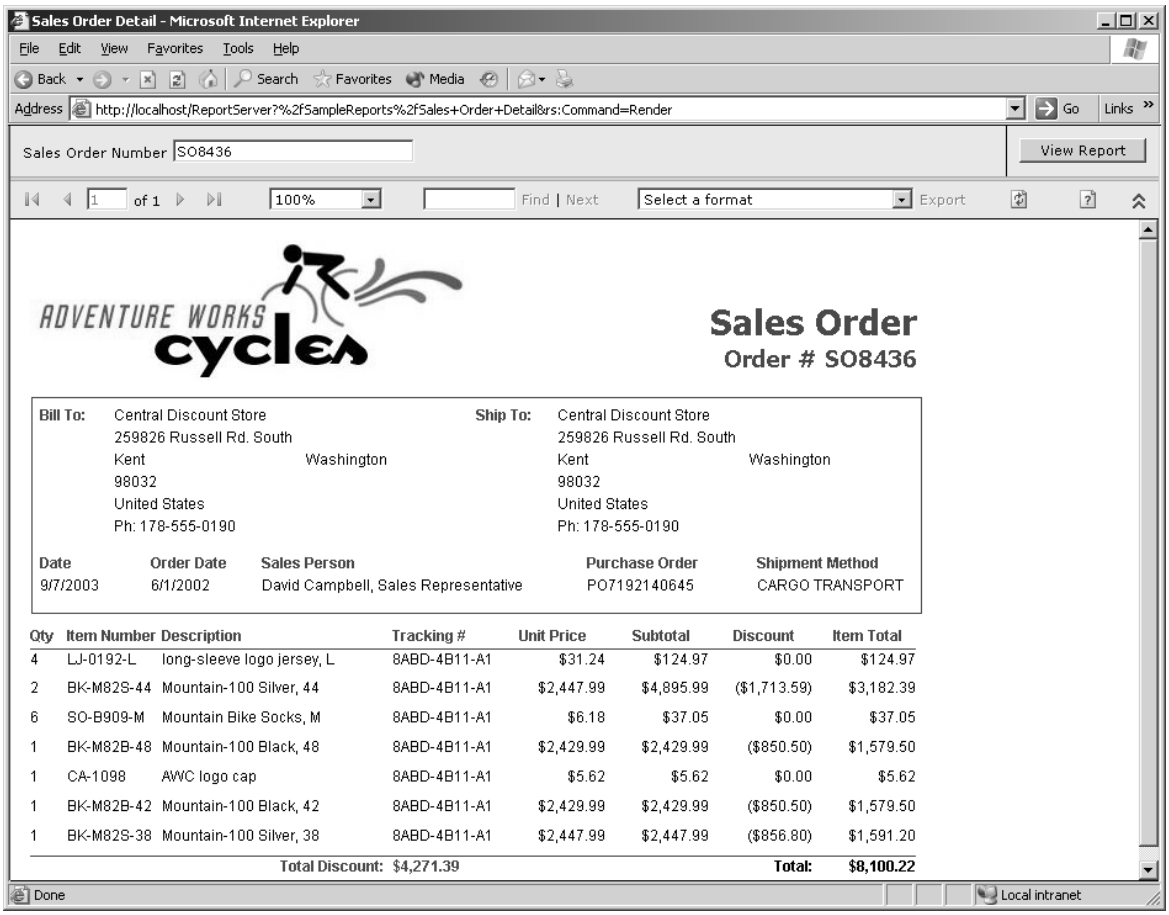
SQL dopyt pre výber údajov v tomto príklade je

```
SELECT      dbo.ProductSubCategory.Name AS ProdSubCat,
            dbo.ProductModel.Name AS ProdModel,
            dbo.ProductCategory.Name AS ProdCat,
            dbo.ProductDescription.Description,
            dbo.ProductPhoto.LargePhoto,
            dbo.Product.Name AS ProdName,
            dbo.Product.ProductNumber,
            dbo.Product.Color,
            dbo.Product.Size,
            dbo.Product.Weight,
            dbo.Product.DealerPrice,
            dbo.Product.Style,
            dbo.Product.Class,
            dbo.Product.ListPrice

FROM  dbo.ProductSubCategory INNER JOIN
      dbo.Locale INNER JOIN
      dbo.ProductDescriptionXLocale ON dbo.Locale.LocaleID = dbo.ProductDescriptionXLocale.LocaleID
         INNER JOIN
      dbo.ProductDescription ON dbo.ProductDescriptionXLocale.ProductDescriptionID =
         dbo.ProductDescription.ProductDescriptionID INNER JOIN
      dbo.ProductModel INNER JOIN
      dbo.Product ON dbo.ProductModel.ProductModelID = dbo.Product.ProductModelID INNER JOIN
      dbo.ProductModelXProductDescriptionXLocale ON
     dbo.ProductModel.ProductModelID = dbo.ProductModelXProductDescriptionXLocale.ProductModelID ON
      dbo.ProductDescriptionXLocale.LocaleID = dbo.ProductModelXProductDescriptionXLocale.LocaleID
         AND
      dbo.ProductDescriptionXLocale.ProductDescriptionID =
         dbo.ProductModelXProductDescriptionXLocale.ProductDescriptionID ON
     dbo.ProductSubCategory.ProductSubCategoryID = dbo.Product.ProductSubCategoryID INNER JOIN
      dbo.ProductCategory ON dbo.ProductSubCategory.ProductCategoryID =
         dbo.ProductCategory.ProductCategoryID LEFT OUTER JOIN
      dbo.ProductPhoto ON dbo.Product.ProductPhotoID = dbo.ProductPhoto.ProductPhotoID

WHERE (dbo.Locale.LocaleID = N'EN')
```

Sales Order Detail



Obr. P.14 Príklad Sales Order Detail – návrh reportu



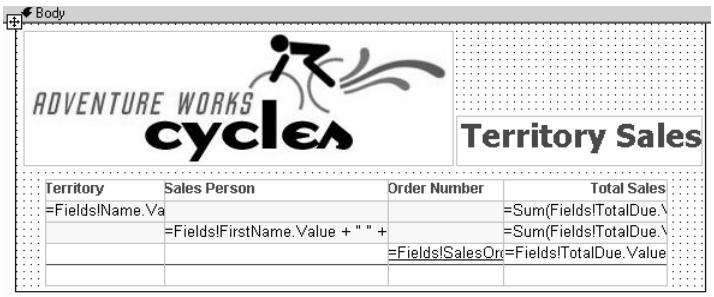
Obr. P.15 Príklad Sales Order Detail – preview reportu

SQL dopyt pre výber údajov v tomto príklade je

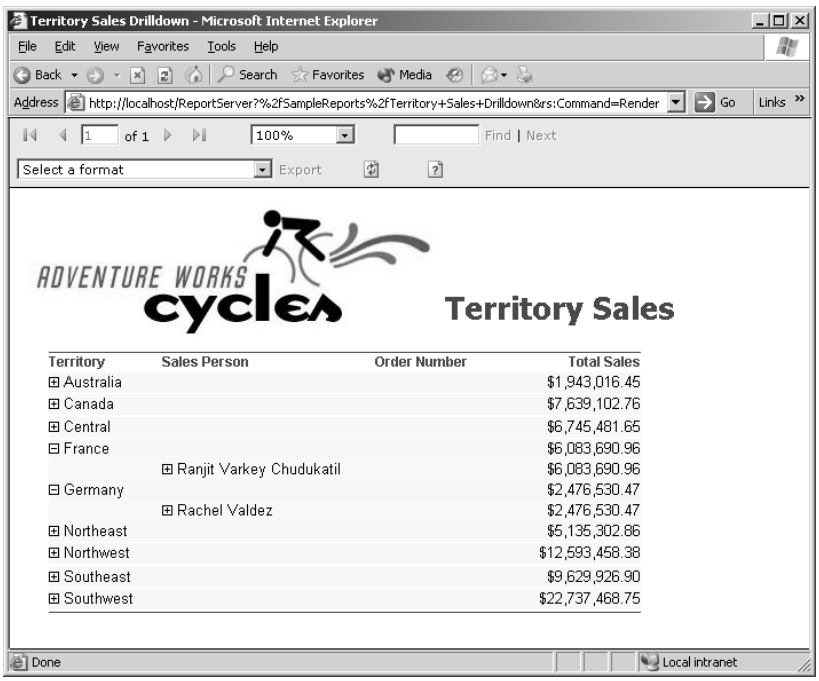
```
SELECT      SalesOrderHeader.SalesOrderNumber, Store.Name AS Store,
            SalesOrderHeader.OrderDate, Employee.FirstName AS SalesFirstName,
            Employee.LastName AS SalesLastName, Employee.Title AS SalesTitle,
            SalesOrderHeader.PurchaseOrderNumber, ShipMethod.Name AS ShipMethod,
            BillAddress.AddressLine1 AS BillAddress1, BillAddress.AddressLine2 AS BillAddress2,
           BillAddress.City AS BillCity,
            BillAddress.PostalCode AS BillPostalCode, BillStateProvince.Name AS BillStateProvince,
```

```
        BillCountryRegion.Name AS BillCountryRegion,
        BillAddress.Phone AS BillPhone, ShipAddress.AddressLine1 AS ShipAddress1,
        ShipAddress.AddressLine2 AS ShipAddress2,
        ShipAddress.City AS ShipCity, ShipAddress.PostalCode AS ShipPostalCode,
        ShipStateProvince.Name AS ShipStateProvince,
        ShipCountryRegion.Name AS ShipCountryRegion, ShipAddress.Phone AS ShipPhone
FROM    SalesOrderHeader LEFT OUTER JOIN
        Address AS BillAddress INNER JOIN
        StateProvince AS BillStateProvince ON BillAddress.StateProvinceID =
            BillStateProvince.StateProvinceID INNER JOIN
        CountryRegion AS BillCountryRegion ON BillAddress.CountryRegionCode =
            BillCountryRegion.CountryRegionCode ON
        SalesOrderHeader.BillToAddressID = BillAddress.AddressID LEFT OUTER JOIN
        Address AS ShipAddress INNER JOIN
        StateProvince AS ShipStateProvince ON ShipAddress.StateProvinceID =
            ShipStateProvince.StateProvinceID INNER JOIN
        CountryRegion AS ShipCountryRegion ON ShipAddress.CountryRegionCode =
            ShipCountryRegion.CountryRegionCode ON
        SalesOrderHeader.ShipToAddressID = ShipAddress.AddressID LEFT OUTER JOIN
        Employee ON SalesOrderHeader.SalesPersonID = Employee.EmployeeID LEFT OUTER JOIN
        ShipMethod ON SalesOrderHeader.ShipMethodID = ShipMethod.ShipMethodID LEFT OUTER JOIN
        Store ON SalesOrderHeader.CustomerID = Store.CustomerID
WHERE   (SalesOrderHeader.SalesOrderNumber = @SalesOrderNumber)
```

Territory Sales



Obr. P.16 Príklad Territory Sales – návrh reportu



Obr. P.17 Príklad Territory Sales – preview

SQL dopyt pre výber údajov v tomto príklade je

```
SELECT      dbo.SalesTerritory.Name, dbo.SalesPerson.SalesPersonID, dbo.Employee.FirstName,
            dbo.Employee.LastName, dbo.SalesOrderHeader.SalesOrderNumber,
            dbo.SalesOrderHeader.TotalDue

FROM   dbo.SalesTerritory INNER JOIN
       dbo.SalesPerson ON dbo.SalesTerritory.TerritoryID = dbo.SalesPerson.TerritoryID INNER JOIN
       dbo.Employee ON dbo.SalesPerson.SalesPersonID = dbo.Employee.EmployeeID INNER JOIN
       dbo.SalesOrderHeader ON dbo.SalesTerritory.TerritoryID = dbo.SalesOrderHeader.TerritoryID AND
       dbo.SalesPerson.SalesPersonID = dbo.SalesOrderHeader.SalesPersonID

GROUP BY    dbo.SalesTerritory.Name, dbo.SalesPerson.SalesPersonID,
            dbo.Employee.FirstName, dbo.Employee.LastName,
            dbo.SalesOrderHeader.SalesOrderNumber, dbo.SalesOrderHeader.TotalDue

ORDER BY dbo.SalesTerritory.Name
```

Materiály na webe

Zdroj	Adresa
Microsoft Support Services	http://support.microsoft.com/directory
Microsoft newsgroups	news://news.microsoft.com/
Microsoft Windows Hardware Compatibility List	http://www.microsoft.com/hcl
Microsoft MSDN®	http://msdn.microsoft.com
Professional Association for SQL Server	http://www.sqlpass.org/
Microsoft SQL Server Developer Center	http://msdn.microsoft.com/sql/
SQL Server Magazine	http://www.sqlmag.com/
Microsoft SQL Server Support	http://support.microsoft.com/support/sql
TechNet site	http://www.microsoft.com/technet/
Microsoft SQL Server	http://www.microsoft.com/sql

Vydal: Microsoft, s.r.o., Novodvorská 1010/14B, 140 00 Praha 4, tel.: +420 261 197 111, fax: +420 260 197 100
http://www.microsoft.com/cze, http://www.microsoft.com/slovakia