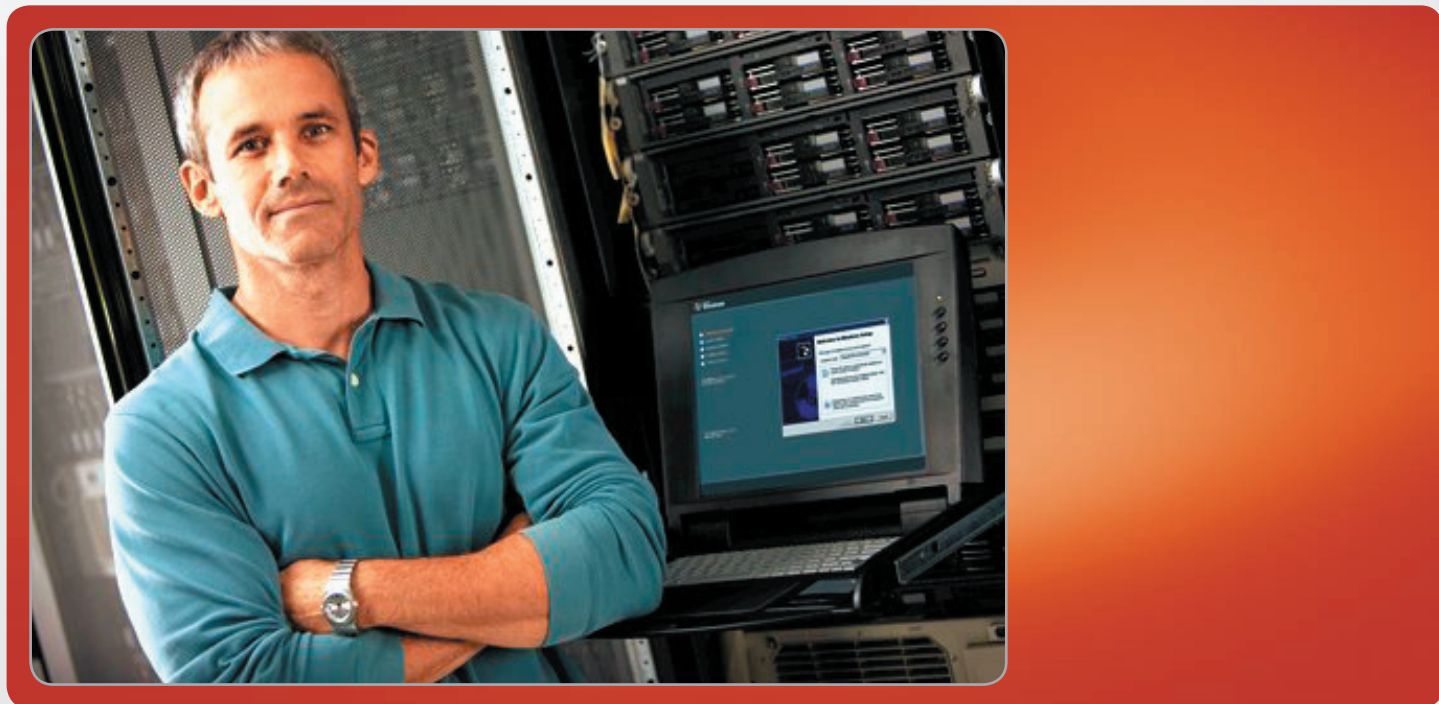


# Business Intelligence v SQL Serveri 2005



Ľuboslav Lacko



# Business Intelligence v SQL Serveri 2005

## Obsah:

Úvod:

Kapitola 1: <b>Microsoft SQL Server 2005 z hľadiska BI</b> .....	2
Kapitola 2: <b>Modelovanie BI projektov v prostredí Visual Studio 2005</b> .....	6
Kapitola 3: <b>Integračné služby</b> .....	11
Kapitola 4: <b>OLAP analýzy</b> .....	30
Kapitola 5: <b>Dolovanie údajov (data mining)</b> .....	50
Kapitola 6: <b>Reportovacie služby</b> .....	73
Kapitola 7: <b>Klientský prístup k analytickým službám</b> .....	96

### *Odporúčanie pre začiatočníkov*

Pre čitateľov, ktorí sa s problematikou integračných služieb, OLAP analýz a data miningu len zoznamujú odporúčame ako prvý krok prejsť si úvodné príklady z kapitol 3, 4, v takomto poradí ako jeden celok, čím získajú základný síce povrchný, ale komplexný prehľad problematiky a až potom študovať jednotlivé kapitoly

## Kapitola 1: Microsoft SQL Server 2005 z hľadiska BI

Poznatky získané z vnútornej činnosti a z interakcie firmy navonok, teda z obchodovania uchované vo formách údajov v databázach sú jedným z najcennejších imaní firiem. Údaje sú spravidla ukladané do transakčných **OLTP** (On-line Transaction Processing) databáz. Tieto sú určené pre vykonávanie veľkého množstva on – line transakcií, napríklad bankových, obchodných a podobne. Takéto databázy sú naviazané na IT systémy, ktorých cieľom je automatizácia každodenných činností, ktoré sú predmetom nášho podnikania, napríklad skladové hospodárstvo, mzdy, nákup a predaj, prípadne riadenie a monitorovanie technologických procesov v reálnom čase. Transakčné systémy sú vo firemnej praxi veľmi často používané a sú veľmi obľúbené, jednak pre svoje výhody, ale aj z dôvodu existencie množstva špecialistov, či už administrátorov, alebo vývojárov. V prípade, ak transakčný databázový systém pokrýva väčšinu podnikových aktivít nazývame ho aj systém ERP (Enterprise Resource Planning). K zdroju údajov teda 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 však optimálne riešenie. Nielen z teórie, ale aj z praktických skúsenosti totiž vyplýva, že údaje v transakčných databázach by mali byť uložené 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 bola preto pomerne neefektívna a značne pomalá. Rozhranie medzi relačnými a analytickými systémami nie je ani ostré, ani jednoznačné. Od klasických transakčných systémov sa plynule dostávame k systémom pre podporu riadenia a rozhodovania alebo inak povedané, z úrovne okamžitých transakcií sa dostávame na akúsi firemnú „operatívnu taktickú úroveň“. Systémy **MIS** (Management Information Systems) do ktorých vstupujú údaje z transakčných systémov 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. Nevýhodou je pomerne veľká réžia. Požiadavky na zostavu sa odoslali vývojovému tímu MIS, ktorý vytvoril zostavu poskytol ju manažérom až po určitom čase, spravidla po niekoľkých dňoch, týždňoch alebo dokonca až po niekoľkých mesiacoch. Operatívnejšie výsledky poskytujú systémy **DSS** – (Decision -Support Systems) kde už v názve je naznačené ich určenie pre podporu rozhodovania. Na rozdiel od systémov typu MIS, ktoré sa nasadzujú na operatívne taktickej úrovni, systémy DSS sú už na rozhraní taktického a strategického rozhodovania. Poskytujú riadiacim pracovníkom napríklad výsledky pomerne zložitých analýz. Cez operačne taktickú úroveň sme sa dostali v našom prehľade až na strategickú úroveň k informačným systémom pre vrcholové riadenie. Tieto sú niekedy označované ako **EIS** – (Executive Information Systems) no oveľa častejšie sa stretáme s termínom Business Intelligence. Tento pojem môžeme definovať ako proces transformácie údajov na informácie prevod týchto informácií na poznatky prostredníctvom objavovania. Účelom Business Intelligence je teda konverzia veľkých objemov údajov na poznatky, ktoré sú potrebné pre koncových používateľov. Tieto poznatky môžeme potom efektívne využiť napríklad v procese rozhodovania. Pod pojmom informácia nerozumieme len konkrétny záznam, alebo množinu záznamov. Často potrebujeme sledovať trend nejakej veličiny, napríklad pri obchodovaní s cennými papiermi, alebo potrebujeme nájsť medzi údajmi určité závislosti. Preto moderné databázové servery obsahujú rozsiahlu podporu pre **OLAP** (Online Analytical Processing), Data Mining (dolovanie, odkrývanie dát) a Data Warehouse (dátové sklady)

### Teoretický úvod do BI

Na úvod je pre prípadných začiatníkov potrebné objasniť pojmy Business Intelligence, OLTP, OLAP a dátový sklad:

**Business Intelligence** je proces transformácie údajov na informácie prevod týchto informácií na poznatky prostredníctvom objavovania. Účelom procesu je konvertovať veľké objemy údajov na poznatky, ktoré sú potrebné pre koncových používateľov. Tieto poznatky môžeme potom efektívne využiť napríklad v procese rozhodovania.

**OLTP** (Online Transaction Processing) – pod touto skratkou sa skrývajú tzv. transakčné databázy, t. j. databázy, ktorých primárnym účelom je vykonávanie veľkého množstva on – line transakcií, napríklad bankových, obchodných a podobne.

**OLAP** (Online Analytical Processing) – táto skratka zahŕňa štruktúry údajov a analytické služby, ktoré slúžia pre analýzu veľkého množstva údajov. Multidimenzionálne OLAP databázy obsahujú fakty vzťahnuté k dimenziám

**Dátový sklad** (*definícia dátového skladu podľa Billa Inmona*) je podnikovo štruktúrovaný depozitár subjektovo orientovaných, integrovaných, časovo premenných, historických dát použitých na získavanie informácií podporu rozhodovania. V dátovom sklade sú uložené atomické a sumárne dáta. Údaje sa získavajú ukladajú do produkčných (operačných) databáz, ktoré môžu byť v rôznych oddeleniach firiem, alebo dokonca v rozličných geografických lokalitách. Tieto údaje v pravidelných intervaloch zozbierame, predspracujeme a zavedieme do dátového skladu.

Dátový sklad je v podstate tiež databáza, len je organizovaná podľa trochu iných pravidiel, tabuľky napríklad nemusia byť normalizované a podobne. Pozrime sa na pojmy z Inmonovej definície trochu podrobnejšie

- **subjektová orientácia:** údaje sa do dátového skladu zapisujú skôr podľa predmetu záujmu, než podľa aplikácie, v ktorej boli vytvorené. Pri orientácii na subjekt sú dáta v dátovom sklade kategorizované podľa subjektu, ktorým môže byť napr. zákazník, dodávateľ, zamestnanec, výrobok a podobne. Orientácia na aplikáciu naproti tomu znamená, že údaje sú v systéme uložené podľa jednotlivých aplikácií, napríklad údaje aplikácie pre odbyt, údaje aplikácie pre fakturáciu, údaje aplikácie pre personalistiku.
- **integrovanosť:** Dátový sklad musí byť jednotný a integrovaný. To znamená, že údaje týkajúce sa konkrétneho predmetu sa do dátového skladu ukladajú len raz. Preto musíme zaviesť jednotnú terminológiu, jednotné a konzistentné jednotky veličín. Nie je to úloha jednoduchá, pretože údaje prichádzajú do dátového skladu z nekonzistentného a neintegrovaného operačného prostredia. Preto musia byť údaje v etape prípravy a zavedenia upravené, vyčistené a zjednotené. Ak údaje nie sú konzistentné a dôveryhodné, dátový sklad stráca význam.
- **časová variabilita:** údaje sa ukladajú do dátového skladu ako séria snímok, z ktorých každá reprezentuje určitý časový úsek. Na rozdiel od operačného prostredia, kde sú údaje platné v okamihu prístupu, v dátových skladoch sú údaje platné pre určitý časový moment, časový snímok. Zatiaľ čo v operačnom databázovom prostredí sa údaje ukladajú za kratšie časové obdobie dní, maximálne mesiacov, v dátovom sklade sú údaje za dlhšie časové obdobie, typicky niekoľko rokov. Kľúčové atribúty v dátovom sklade obsahujú čas, ktorý v operačných databázach nemusí byť uvádzaný. Akonáhle je v dátovom sklade zaznamenaná konkrétna snímka dát z operatívnej databázy, nemôžu byť už tieto údaje v dátovom sklade modifikované.
- **nemennosť:** V operačných transakčných databázach sú údaje do databázy jednak vkladané, jednak modifikované a aj mazané. Údaje v dátovom sklade sa obvykle nemenia ani neodstraňujú, len sa v pravidelných intervaloch pridávajú nové údaje. Preto je manipulácia s údajmi ďaleko jednoduchšia v dátových skladoch. V zásade môžeme pripustiť len dva typy operácií. Zavedenie údajov do dátového skladu a prístup k týmto údajom. žiadne zmeny údajov nie sú prípustné. Z toho vyplýva, že väčšina metód pre optimalizáciu a normalizáciu údajov a transakčný prístup k údajom je v dátovom sklade nepotrebná.

### Kvalita údajov pre analýzy

Firmy využívajú pre svoju činnosť rôzne druhy ekonomického softwaru, napríklad účtovníctvo, skladové hospodárstvo, evidencia pohybu tovaru a podobne, pričom samozrejme zhromažďujú údaje, zčasti sú možno bezcenné ale možno aj veľmi cenné, tieto však zostávajú nevyužitú, pretože sú uložené vo forme, ktorá ich robí nedostupnými pre účely získavania informácií. Existencia údajov totiž vôbec neznamena dostupnosť informácií. Predstavme si firmu čo i len strednej veľkosti, ktorá predáva 1000 druhov tovarov, prostredníctvom 10 predajných kanálov, 100 odberateľom. Ľahko sa dopočítame milióna možných kombinácií spomínaných položiek. A keby sme chceli sledovať obchodný život firmy po mesiacoch, dostávame dvanásť miliónov možných kombinácií. Ak by sme chceli sledovať viac ukazovateľov, napríklad zisky, výsledky kampanie podobne museli by sme spomínaných dvanásť miliónov ešte prenásobiť počtom ukazovateľov.

### Polemika okolo využívania transakčných databáz pre analýzy

Transakčné OLTP databázy sú určené pre ukladanie operačných údajov. Výsledkom dopytovania sú databázové tabuľky, súhrny získané pomocou agregáčnych funkcií, rôzne zostavy a podobne. OLTP databázy sú z dôvodu jednoduchého dotazovania a vylúčenia redundancie spravidla normalizované. Takéto systémy dosahujú vysoké výkony skôr pri on-line transakciách než pri zložitých analýzách, ktoré sú veľmi náročné na výpočtovú kapacitu. Komplexná analýza vyžaduje iné techniky návrhu databáz, napríklad použitie multidimenzionálnych hviezdových schém s tabuľkami faktov, ktoré obsahujú merné jednotky obchodovania a vysoko nenormalizované tabuľky dimenzií. Azda najväčšou prekážkou použitia databázových systémov OLTP pre analýzy je skutočnosť, že tieto systémy nemajú k dispozícii integrovaný zdroj údajov zo všetkých operačných systémov v rámci podniku tak, aby umožnili tvorbu komplexných analýz, to znamená, že potrebné údaje, alebo údaje ktoré by mali slúžiť ako podklady pre analýzy sú roztrúsené v rôznych spravidla heterogénnych OLTP systémoch a musia sa zakaždým pracne integrovať skôr, ako je možné získať požadované informácie. Časová náročnosť prípadných analýz, hoci nemusí ísť ani o príliš zložitú ani príliš komplexnú analýzu je preto pomerne vysoká. Niekedy sa dokonca ani nepodarí koordinovať dáta medzi jednotlivými systémami takže vlastne ani nemôžeme získať globálny obraz o stave podnikania.

**Nevýhody** použitia transakčných databáz pre analytické účely by sme mohli zhrnúť do niekoľkých bodov

- nie je jednoduché nájsť príčiny vysvetlenia problémov
- obtiažne hľadanie závislosti jednotlivých veličín
- príliš rozsiahle výstupy z transakčných systémov
- dlhý čas výpočtu a degradácia výpočtového výkonu databázového stroja transakčnej databázy
- transakčný systém neuchováva historické údaje
- nehomogénna štruktúra údajov
- dlhý čas prípravy údajov

### Výhody

Na druhej strane je snaha o komplexný prístup k údajom v dátovom sklade pri akceptovaní hlavnej požiadavky, ktorú prináša dnešná hektická globalizovaná ekonomika – „všetko v reálnom čase“. Kým v minulosti sa údaje z dátových skladov spracovávali v dávkach pre nejaké obdobie z blízkej minulosti (včerašší deň, minulý týždeň), v súčasnosti manažéri a analytici požadujú prístup k výsledkom analýz v reálnom čase. V návaznosti na túto požiadavku už začína byť minulosťou aj oddelenie transakčných systémov a dátových skladov. Súčasným trendom vyplývajúcim z citovaných požiadaviek je viacúrovňový podnikový dátový sklad s možnosťou prístupu pomocou integrovaného aplikačného prostredia. Vznikne tak akýsi komplexný BI ekosystém, do ktorého sú na strane vstupov spravidla zahrnuté aj externé zdroje údajov a ktorý na výstupe poskytuje informácie nielen pre firmu, ktorá je vlastníkom, budovateľom a prevádzkovateľom BI ekosystému, ale aj pre jej partnerov, dodávateľov a zákazníkov.

### BI systémy pracujúce v reálnom čase

BI systémy pracujúce v reálnom čase sú založené na troch primárnych faktoroch

- paralelný priebeh etapy ETL, teda extrakcie, transformácie a zavedenia údajov, pričom všetky ETL procesy musia byť dostatočne nadimenzované na predpokladané objemy spracovávaných údajov a to aj v dobe špičiek, ako sú napríklad predvianočný predaj, koncoročné uzávierky alebo marketingové kampane
- relačné systémy musia okrem operačnej prevádzky, teda spracovávania transakcií zvládnuť aj zložité a kapacitne náročné BI dopyty
- doručovanie výsledkov BI procesov, teda reportov a výsledkov analýz a to buď periodicky, alebo na vyžiadanie

Charakteristickou črtou dátových skladov je veľké množstvo údajov v niektorých databázových tabuľkách.

V takýchto prípadoch systémy reálneho času je výhodné rozdeliť veľkú tabuľku na niekoľko logických oddielov (partícií), podľa nejakého pravidla vyplývajúceho z povahy údajov, napríklad jeden logický oddiel môže obsahovať údaje za nejaké časové obdobie (mesiac, rok), prípadne údaje podľa iného pravidla, napríklad v zozname osôb, osoby, ktorých priezvisko začína príslušným písmenom a podobne. Takéto delenie je potom logické aj z hľadiska následného spracovania údajov, pretože trebárs reklamácie a fakturácie sa obvykle vykonávajú za príslušné kalendárne obdobie

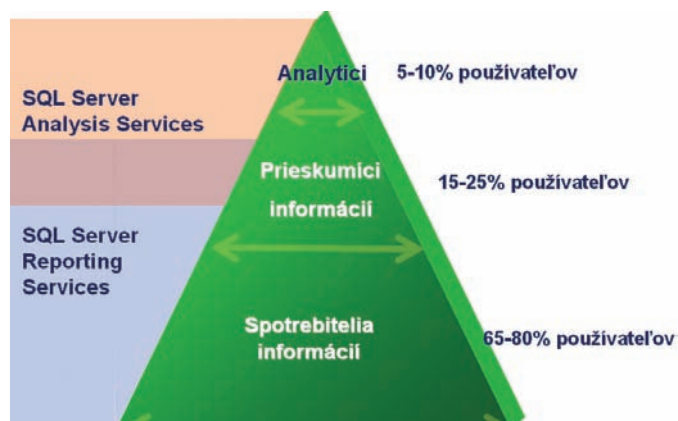
### Implementácia BI v SQL Serveri 2005

Ak porovnáme analytické služby SQL nového Serveru 2005 s momentálne komerčne dostupnou verziou SQL Server 2000, nájdeme mnohé vylepšenia návrhového prostredia, no najvýznamnejší rozdiel je vo filozofii.



*Náročnosť použitia jednotlivých business intelligence technológií*

Podľa rôznych prieskumov 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.



Percentuálne využitie jednotlivých business intelligence technológií

Na podnikovej úrovni sa generujú rôzne druhy 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 prístupujú v rámci podnikového Intranetu.

Funkcia	Standard	Enterprise	Poznámky
Data Warehousing	✓	✓	
BI Development Studio	✓	✓	Integrované vývojové nástroje pre tvorbu a ladenie dátovej integrácie, OLAP, data miningu a reportovacích riešení
Analysis Services (OLAP Engine)	✓	✓	Obsahuje rozšírené OLAP funkcie, vrátane KPI
Partitioned Cubes		✓	
Proactive Caching		✓	Poskytuje automatizovaný caching pre väčšiu škálovateľnosť a výkon
Advanced Measures and Dimensions		✓	
Custom Rollups		✓	
Parallel Data Modeling		✓	
Data Mining—štandardné algoritmy	✓	✓	Rozhodovacie stromy (decision trees) a zhľukovanie (clustering)
Data Mining—rozšírené algoritmy	✓	✓	Ďalšie algoritmy (neurónové siete, na?ve bayes, časové rady, asociácie, clustering sekvencií,...
Reporting Controls and Wizard	✓	✓	Ovládacie prvky pre prácu s reportmi pre webové aj Windows aplikácie sú súčasťou VS 2005
Reporting Engine	✓	✓	
Data-Driven Subscriptions		✓	Podpora rozsiahlej distribúcie používateľsky definovaných reportov
Scale-out Web Farms		✓	
Infinite Drilldown		✓	
Report Builder	✓	✓	Nástroj pre koncových používateľov umožňujúci tvorbu vlastných reportov

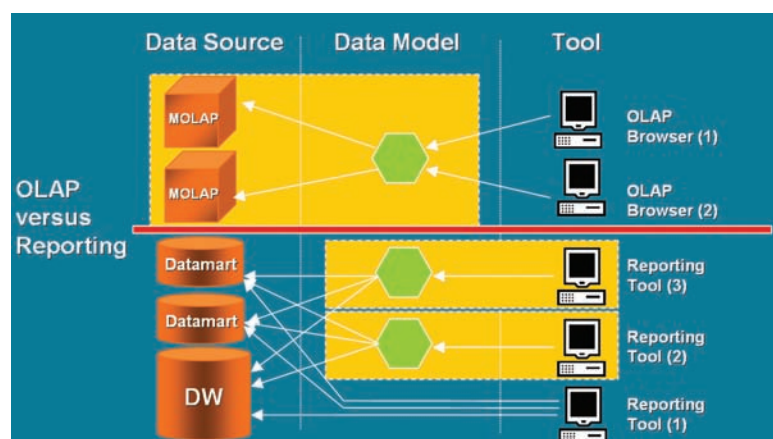
## Kapitola 2: Modelovanie BI projektov v prostredí Visual Studio 2005

Jedným z preferovaných trendov pre budovanie podnikových projektov aplikácií je model driven development, kedy sa budovanie projektu začne fázou modelovania. Túto filozofiu podporuje aj SQL Server 2005, ktorý pre modelovanie projektov typu Business Intelligence využíva technológiu UDM (Unified Dimension Model). Na základe vytvoreného modelu BI aplikácie bude následne vygenerovaná štruktúra pre dátový sklad, vytvorené dávky pre jeho plnenie prostredníctvom možností Integrovaných služieb. Budovanie BI projektu pokračuje návrhom mierok, dimenzií, kociek a podobne. U SQL Servera 2005 slúži pre modelovanie grafické návrhové prostredie Visual Studio 2005

Prvý predpoklad pre unifikovaný prístup k modelovaniu je unifikácia technológií. Tento predpoklad je splnený integráciou databázových analytických a reportovacích služieb do jedného balíka. Druhým predpokladom je unifikované používateľské rozhranie, čo v prípade databázového servera predstavujú nástroje pre administráciu, dopytovanie a návrh štruktúr a čiastkových modelov. Aj táto požiadavka je v prípade SQL Servera 2005 splnená. K dispozícii je Visual Studio 2005, pričom interakcia medzi SQL Serverom 2005 a vývojovým prostredím Visual Studio 2005 je veľmi tesná. Verzia Visual Studia pre BI projekty sa nainštaluje spolu s SQL Serverom 2005

### Základné princípy UDM

U predchádzajúcej verzii SQL Servera 2000 sú vrstvy databázových, analytických a reportovacích služieb pomerne striktné oddelené, aj keď aj tu už nájdeme množstvo zjednocujúcich prvkov – je možné analyzovať údaje z relačných databáz, prípadne generovať reporty z relačných aj analytických databáz. Prípadne pomocou MDX dotazu (Skratka MDX je akronym od slovného spojenia **M**ultidimensional **E**xpresions – multidimenzionálne výrazy) vyšpecifikujeme určitú podmnožinu údajov z multidimenzionálnej štruktúry (OLAP kocky) a vypíšeme ju do dvojzmernej tabuľky, ktorá obsahuje množinu buniek. A dvojzmerne tabuľky – na to sa už predsa ideálne hodia reportovacie služby.



Oddelenie vrstvy OLAP a reportovacích služieb u SQL Serveru 2000

Hlavná nevýhoda architektúry predchádzajúcej verzie – redundancia údajov a modelov je zrejmá už na prvý pohľad. Na obrázku v ľavej časti vidíme už na prvý pohľad, že dochádza k redundancii údajov, pretože tie isté údaje sú jednak v relačných a jednak v multidimenzionálnych databázach. Pri hlbšej analýze dátových tokov zistíme, že situácia z hľadiska redundancie je ešte komplikovanejšia. Tie isté údaje sa prenášajú z relačných databáz do dátových skladov, odtiaľ v mnohých prípadoch do dátových tržníc a odtiaľ do multidimenzionálnych OLAP kociek. Taktiež nezanedbateľná redundancia a s ňou spojené množstvo práce vysokokvalifikovaných špecialistov je vo vrstve modelov. Iným problémom takejto oddelenej architektúry sú rozdielne návyky pracovníkov čo sa týka práce s údajmi. Problematika preorientovania sa z relačných databáz na OLAP kocky.

### Relačné databázy

- Viacnásobné tabuľky faktov
- Žiadne obmedzenia pre atribúty dimenzií
- Možnosť transakčných prístupov
- Možnosť použitia vyšších normálnych foriem
- Relačné vzťahy medzi objektami v databáze
- Možnosť definovania zložitých vzťahov typu many-to-many

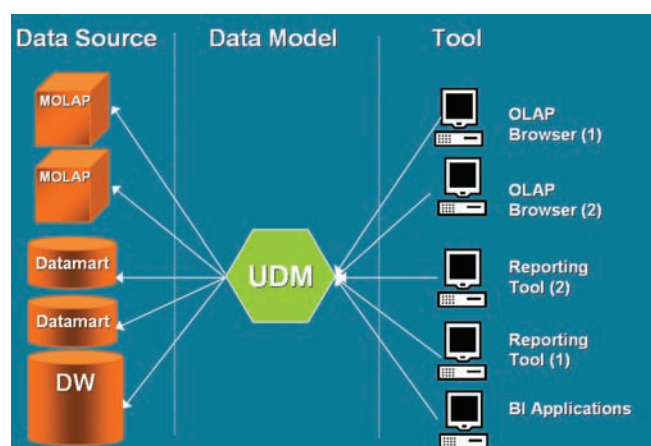
### Multidimenzionálne databázy

- Multidimenzionálna navigácia
- Možnosť hierarchickej prezentácie
- Mená entít sa vzťahujú k predmetu podnikania
- Možnosť výkonných MDX kalkulácií
- Pohľad na údaje z viacerých perspektív
- Partície
- Databáza obsahuje výsledky agregácií
- Distribuované zdroje

Modelovanie s využitím UDM umožňuje využitie výhod oboch typov databáz a do istej miery eliminuje ich nevýhody. Oblasť výhodnosti použitia reportov z relačných a OLAP databáz naznačuje aj tabuľka

Vlastnosť	Relačné DB	OLAP
Pružné schémy	✓	✗
Prístup k údajom v reálnom čase	✓	✗
Jednotné úložisko údajov	✓	✗
Jednoduchá správa	✓	✗
Detailné reportovanie	✓	✗
Vysoký výkon	✗	✓
Orientácia na koncového používateľa	✗	✓
Jednoduchá navigácia	✗	✓
Zložité analýzy	✗	✓

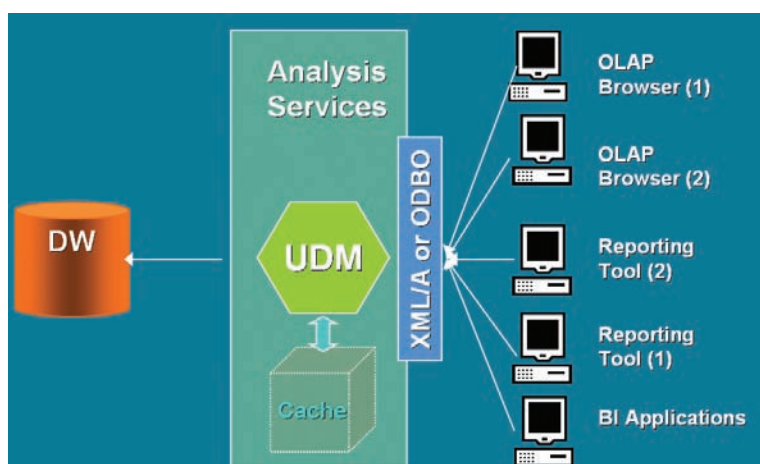
V novej verzii SQL Servera 2005 sú vrstvy Business Intelligence zjednotené do jednotného modelu Unified Dimensional Model (UDM). Tento model prevzal to najlepšie z reportovania a OLAP analýz. K zjednoteniu dochádza jednak na úrovni modelov, kedy je potrebný len jeden dimenzionálny model pre generovanie reportov aj OLAP kociek.



Unified Dimensional Model – zjednotenie na úrovni modelov



Ak sa čitatelia majúci skúsenosti zo staršou verziou SQL Servera 2000 (prípadne iných moderných databázových serverov obsahujúcich analytické a reportovacie služby, napríklad Oracle 10g, alebo IBM DB2 Stinger) zamyslia nad príkladmi analýz, ktoré je možné pomocou týchto nástrojov vytvárať, vždy sa vybrali tabuľky z relačnej databázy alebo z dátového skladu, ktoré slúžili pre vytvorenie faktov a dimenzií. Mechanizmus MOLAP (multidimenzionálne on-line analytické spracovanie) potom uloží analytické dáta vo vlastných dátových štruktúrach a sumároch. Počas tohto procesu sa napočíta toľko predbežných výsledkov, koľko je z technického a časového hľadiska možné. Údaje v úložisku typu MOLAP sa teda budú ukladať ako vopred vypočítané pole. Hodnoty dát aj indexov sa uchovávajú v jednotlivých poliach multidimenzionálnej databázy. Databáza je organizovaná tak, aby umožnila rýchle získavanie príslušných údajov z viacerých dimenzií. Preto hlavnou výhodou MOLAP je maximálny výkon vzhľadom na dopyty používateľa, nevýhodou je redundancia údajov, nakoľko tieto sú uložené jednak v relačnej databáze, jednak v multidimenzionálnej databáze. Požiadavky na úložnú kapacitu môžu v prípade použitia viacerých dimenzií extrémne narastať. Východiskom z tejto situácie u SQL Servera 2005 je vyrovnávací MOLAP cache pamäť, kam sa ukladajú najčastejšie používané alebo očakávané výsledky agregácií. Navyše toto proaktívne cachovanie je u SQL Servera 2005 plne automatizované. Údaje teda zostávajú v relačných databázach a napočítané agregácie sa ukladajú do multidimenzionálnych štruktúr. Pri dopytovaní sa údaje vyberajú do multidimenzionálnej pamäti cache.



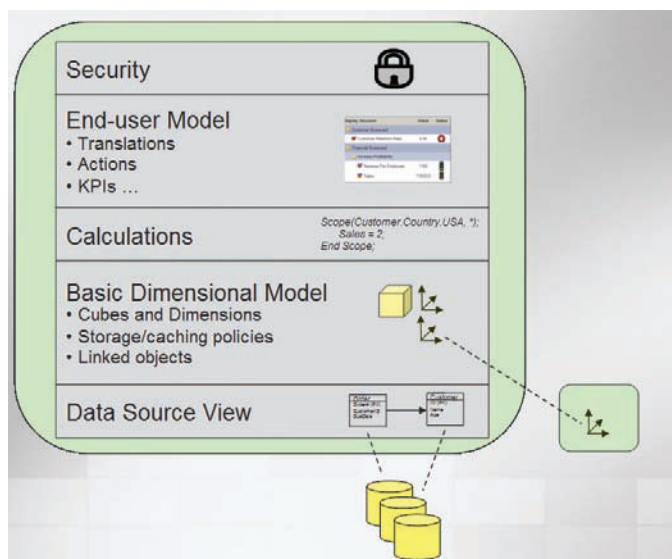
Unified Dimensional Model – cache

Významnou novinkou je zavedenie novej skupiny služieb s názvom Integration Services. Na úrovni tejto vrstvy sa budú integrovať údaje z rôznych dátových zdrojov vrátane ich požadovaných transformácií.

Ak by sme mali zhrnúť dosiaľ prezentované fakty do jednoduchšej definície UDM, dospejeme k záveru že UDM je akýmsi pomysleným mostom medzi používateľom a jeho údajmi.

## Implementácia a prístup k UDM cez Business Intelligence projekty Visual Studia 2005

Azda najlepšie pochopíme filozofiu modelovania na hierarchickej schéme. Celá vrstva slúžiaca pre vytvorenie výstupu (reportu) je z architektonického hľadiska postavená nad údajmi v databázach. Samozrejme je výhodné, aby tieto údaje boli čo najviac vyčistené, prípadne do rozumnej miery normalizované bez zbytočných redundancií a neprehľadných komplikovaných relačných vzťahov. Samozrejme nie každý deň sú Vianoce a tak často pri budovaní BI vrstvy musíme vychádzať z údajov aké sú k dispozícii, prípadne aké vyhovujú logike predmetu podnikania, alebo k akej štruktúre údajov sme sa dopracovali postupným budovaním informačných systémov.



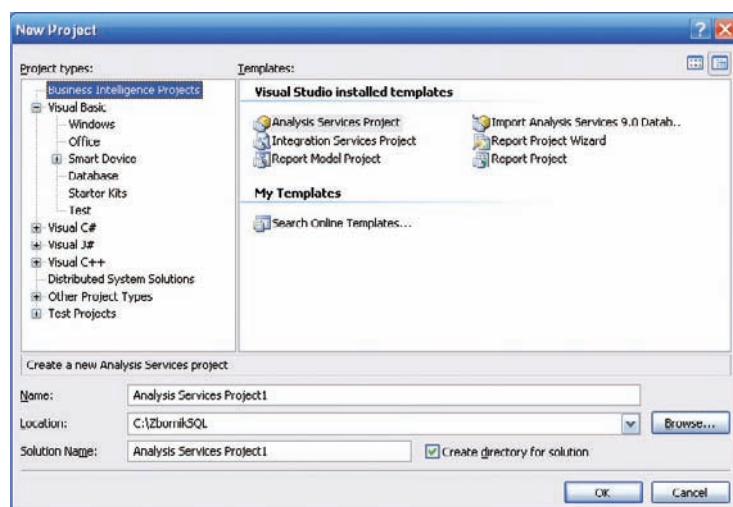
*Unified Dimensional Model*

Samotná BI nadstavba nad databázou je rozdelená do štyroch vrstiev. Nad databázami je vrstva dátových pohľadov, v ktorej už presne vyšpecifikujeme tabuľky, pohľady a to až na úroveň jednotlivých atribútov a relačných väzieb. Modelovaním vrstvy **Data Source View** do určitej miery prispejeme síce nie k fyzickému vyčisteniu údajov no k nadefinovaniu priamejšieho prístupu k nim. Zjednodušene povedané – na tejto úrovni vytvárame relačné schémy na základe ktorých budú vytvárané dimenzie a kocky. Veľmi dôležitou úlohou úrovne Data Source View je zjednotenie údajov z heterogénnych zdrojov.

Nasleduje úroveň dimenzií, kde špecifikujeme ktoré atribúty sa viažu k merným jednotkám obchodovania. Na úrovni **základného dimenzionálneho modelu** do značnej miery ovplyvňujeme aj spôsob ukladania a cachovania multidimenzionálnych údajov. Okrem faktov a dimenzií vstupujú do hry aj **kalkulácie**, to znamená údaje vypočítané podľa jednoduchých, prípadne aj zložitých vzťahov z hodnôt iných atribútova mierok. Napríklad ak by sme mali v tabuľke ako atribút rodné číslo, dokážeme z neho určiť dátum narodenia a teda aj vek príslušnej osoby.

Vrstva **End-user Model** definuje manipulácie s údajmi a výsledkami súhrnov na najvyššej úrovni. Je potrebné si uvedomiť, že výsledky analýz a agregované údaje obsahujú oveľa citlivejšie informácie, než surové údaje. Preto na úrovni výsledkov analýz je oveľa dôležitejšie zabezpečenie cieľeného prístupu k informáciám.

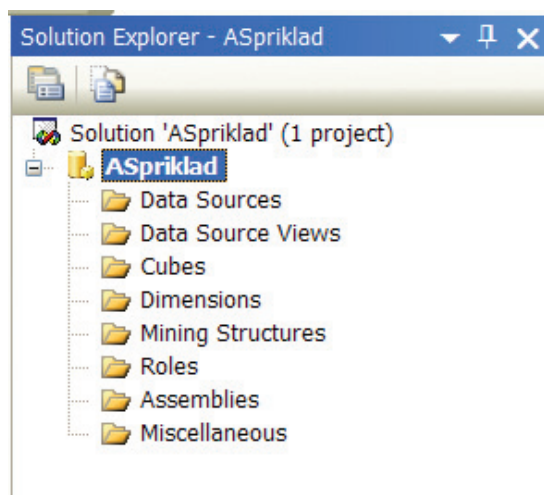
Pre vytváranie analytických projektov je prispôsobená aj koncepcia Business Intelligence projektov vo vývojovom prostredí Visual Studio 2005.



*Ponúkané šablóny typov Business Intelligence projektov vo vývojovom prostredí Visual Studio 2005*

Unifikovaný prístup k BI úlohám podľa tejto viacvrstvovej schémy je naznačený aj zložkami v okne vývojového prostredia Solution Explorer v pravom hornom rohu.

- Data Sources
- Data Source Views
- Cubes
- Dimension
- Mining Models
- Roles
- Assemblies
- Miscellaneous



*Visual Studio 2005 v režime Business Intelligence projektu – okno Solution Explorer*

Pri vytváraní modelu vlastne s väčšou alebo menšou pomocou interaktívnych sprievodcov budujeme v záložkách jednotlivé hierarchické vrstvy modelu.

## Kapitola 3: Integrované služby SQL Servera 2005

S informačnými systémami úzko súvisí nielen ukladanie údajov, ale aj ich zber, presuny, import a export. Túto problematiku musíme riešiť takmer u každého databázového systému. Fáza zavádzania údajov je taktiež neodmysliteľnou súčasťou každého dátového skladu.

Pokúsime sa predstaviť najčastejšie v praxi sa vyskytujúce scenáre a naznačiť možnosti ich riešenia.

- inštalácia informačného systému, alebo jeho databázovej časti
- migrácia na iný informačný systém
- výmena databázového servera
- zavádzanie údajov do dátového skladu

Ak sa na vymenované scenáre pozrieme bližšie, nájdeme určité podobnosti. Pri inštalácii nového informačného systému, alebo dátového skladu sú údaje, ktoré chceme do tohto systému alebo skladu umiestnené spravidla v rôznych nehomogénnych operačných prostrediach, hardvérových platformách (PC, mainframe, iMac...), operačných systémoch (Windows, Unix, Linux, Sun, Solaris...), databázových systémoch (SQL Server, Oracle, Informix, IBM DB2...) archívnych systémoch, podnikových systémoch (SAP, PeopleSoft, Baan...) a čo je ešte horšie, môžu sa vyskytovať v rozličných formátoch. Z predchádzajúcej vety je zrejmé, že do hry vstúpili čiastočne aj zvyšné dva scenáre, teda migrácia na iný informačný alebo databázový systém.

Hlavným rozdielom medzi importom údajov do informačných systémov a zavádzaním údajov do dátového skladu spočíva v tom, že import je záležitosťou jednorazovou, kdežto zavádzanie údajov do dátového skladu sa odohráva periodicky v určitých, napríklad 24 hodinových intervaloch. Aj u týchto dvoch scenárov je začiatok veľmi podobný, import údajov do informačného systému a prvotné naplnenie dátového skladu. A aby to nebolo také jednoduché, nemôžeme nespomenúť archívne dáta obsahujúce historické údaje. Hlavný rozdiel medzi dátovým skladoom a archívom je v tom, že údaje v dátovom sklade sa pravidelne obnovujú. Údaje z archívov sú nezastupiteľným zdrojom historických dát pri prvotnom naplnení dátového skladu.

### Import – Export

Import a export údajov sa vo väčšine prípadov javia ako spojené nádoby a to o akú operáciu v konkrétnom prípade ide, závisí od uhľa pohľadu. Ak máme zámysel importovať údaje do databázy v nejakom formáte, je potrebné na druhej strane v „zdrojovom“ systéme zariadiť ich export. Pomocou takejto nepriamej metódy, ak využijeme napríklad textový formát s údajmi oddelenými čiarkou dokážeme importovať a exportovať údaje prakticky medzi akýmkoľvek heterogénnymi systémami.

Ak exportujeme údaje z databázového systému (Informix, MySQL...) alebo z tabuľkového procesora (Excel) spravidla je možné formát údajov definovať, napríklad ako textový súbor, kde bude každý záznam v jednom riadku, v ňom sú jednotlivé atribúty (hodnoty stĺpcov) oddelené čiarkami. Napriek tomu, že v našom príklade výpisu zaberie každý záznam tri riadky, je to spôsobené len zarovnaním riadkov v textovom editore. Na konci každého záznamu sú neviditeľné hodnoty pre odriadkovanie. Sú to znaky CR a LF (hexadecimálne 0D, 0A )

```
ProspectAlternateKey,FirstName,MiddleInitial,LastName,BirthDate,MaritalStatus,Gender,EmailAddress,YearlyIncome,TotalChildren,NumberChildrenAtHome,Education,Occupation,HouseOwnerFlag,NumberCarsOwned,AddressLine1,AddressLine2,City,State,ZIP,Phone
8725,Stacy,P,Sanz,2/22/1950 0:00,M,F,ssanz@humongousinsurance.com,50000,2,1,Partial
Co,Professional,1,2,3828 Baltic Sea Ct,,Lake Oswego,OR,97034,497-555-0111
57168099284,Trevor,L,Griffin,1/27/1951 0:00,M,M,tgriffin@humongousinsurance.
com,50000,2,1,High Schoo,Professional,1,2,3947 Alhambra Valley Rd,,Lake
Oswego,OR,97034,872-555-0163
```

V predchádzajúcom výpise, hlavne ak je k dispozícii záhlavie, je štruktúra údajov zrejماً a pri znalosti problematiky by si skúsený odborník vedel pri importe údajov z takto technokraticky organizovaného zdroja údajov vedel poradiť aj bez záhlavia.

Ako ďalší príklad zdroja pre import údajov použijeme zoznam prezidentov a viceprezidentov USA vrátane rokov vymedzujúcich ich úradovanie, teda text, ktorý ľahko získame z Internetu v stovkách rôznych variánt. Nie je to už presne organizovaný flat súbor, ale pomerne voľne organizovaný textový dokument, ktorý obsahuje všetky

požadované údaje vo voľnej forme. Aby sme výpisom neoslovili len znalcov histórie USA uvádzame koniec zoznamu, teda prezidentov z nedávnej minulosti

```
Richard Nixon (1969-1974) Spiro Agnew (1969-1973) none (1973) Gerald Ford (1973-
1974)
Gerald Ford (1974-1977) none (1974) Nelson Rockefeller (1974-1977)
Jimmy Carter (1977-1981) Walter Mondale (1977-1981)
Ronald Reagan (1981-1989) George Bush (1981-1989)
George Bush (1989-1993) Dan Quayle (1989-1993)
Bill Clinton (1993-2001) Al Gore (1993-2001)
George W. Bush (2001- 2004) Dick Cheney (2001- 2004)
...
```

V zozname vidíme rôzne anomálie, napríklad v rámci funkčného obdobia jedného prezidenta sa vystriedalo viac viceprezidentov a v dvoch krátkych obdobiach rokov 1973 a 1974 funkcia viceprezidenta obsadená nebola. Zdá sa vám to trochu neprehľadné? V tomto prípade s databázou prezidentov by sme čo sa týka zdroja údajov obišli ešte pomerne dobre. Skúste aspoň približne odhadnúť, čo vyjadrujú tieto údaje

```
p,x,s,n,t,p,f,c,n,k,e,e,s,s,w,w,p,w,o,p,k,s,u
e,x,s,y,t,a,f,c,b,k,e,c,s,s,w,w,p,w,o,p,n,n,g
e,b,s,w,t,l,f,c,b,n,e,c,s,s,w,w,p,w,o,p,n,n,m
...
```

Môže to byť doslova čokoľvek, ale aby sme vás dlho nenapínali, jedná sa o databázu húb. Zdrojom je *Department of Information and Computer Science University of California, Irvine CA*. (<http://www.ics.uci.edu/~mlearn/MLRepository.html>)

Každý riadok predstavuje záznam o jednej hube. Nakoľko sa jedná o ukážkovú databázu pre data mining, prvý atribút, ktorým je názov bol odobratý. Jednotlivé písmenká znamenajú vlastnosti príslušnej huby

```
Poživatelnosť : edible=e, poisonous=p
Tvar klobúku: bell=b,conical=c,convex=x,flat=f, knobbed=k,sunken=s
Povrch klobúku: fibrous=f,grooves=g,scaly=y,smooth=s
Farba klobúku: brown=n,buff=b,cinnamon=c,gray=g,green=r,
pink=p,purple=u,red=e,white=w,yellow=y
...
```

To znamená, že k takto koncipovanej databáze je potrebné mať k dispozícii aj legendu, spravidla v tvare definičného súboru. Ak chceme získať použiteľnú a zrozumiteľnú databázu, je potrebné údaje pretransformovať do približne takéhoto tvaru

id	Pozivatelnosť	TvarKlobu ku	PovrchKlob	FarbaKlob	Ox.ZmenaBarvy	Zapach
1	nejedlá, jedovatá	vypuklý	hladký	hnedá	ano	štiplavý, čpavý
3	jedlá	zvonovitý	hladký	biela	ano	anýzový
5	jedlá	vypouklý	hladký	šedá	nie	žiadny

Každý si dokáže urobiť približnú predstavu o náročnosti importu údajov z takéhoto zdroja. Mohli by sme do článku vložiť postup, ako túto databázu pretransformovať zo zdrojového do cieľového tvaru, no nemá to príliš veľký význam. Pri importe údajov je prípad od prípadu úplne iný, aj keď určité vzory a princípy sa používajú dosť často. Napríklad tieto údaje

```
6 0 66 50 1
6 1 70 50 2
6 0 69 50 3
6 0 68 100 4
6 0 67 200 5
...
```

Môžu znamenať čokoľvek. Môžu to byť záznamy z meraní, výsledky analýzy medicínskych vzorkov, ...

Až legenda nám objasní, že ide o záznamy z meraní tesniacich krúžkov raketoplánu, čiže záležitosť, na ktorej záviseli životy mnohých ľudí

1. Number of O-rings at risk on a given flight
2. Number experiencing thermal distress
3. Launch temperature (degrees F)
4. Leak-check pressure (psi)
5. Temporal order of flight

### Externé údaje

Okrem interných údajov z nášho podnikateľského prostredia je pomerne často potrebné pracovať aj s externými údajmi. Tieto údaje môžeme získať napríklad analýzou konkurenčného prostredia, zakúpením údajov o zákazníkoch, alebo využívať môžeme aj údaje voľne prístupné na Internete. Nevýhodou vyplývajúcou so samotnej povahy externých údajov je, že nemôžeme periodicky odoberať vzorky, ako je tomu u interných údajov. Zdroje externých údajov preto vyžadujú nepretržité monitorovanie za účelom určenia, kedy sú dostupné.

### Transformácia a čistenie údajov

V závažných prípadoch, ak sa údaje používajú pre podporu rozhodovania v zložitých veciach je mnohokrát lepšie nemať žiadne údaje, než mať údaje nekvalitné. Znižuje to jednak dôveru v riešenie informačného systému, alebo dátového skladu a takéto údaje môžu viesť k chybným rozhodnutiam. Jadro problému je v tom, že v zdrojových systémoch sa dosť často vyskytujú nekvalitné, alebo nepresné údaje.

Má to aj praktický význam. Napríklad chceme na základe data miningových analýz osloviť marketingovou kampaňou určitú skupinu zákazníkov a chceme to realizovať formou poštových zásielok. Ak z niektorej pobočky alebo od obchodného zástupcu získame adresy zákazníkov v takejto podobe

name	address1
Nowmer Sheri	Bailey Road 2433 Tlaxiaco 15057 Mexico
Whelply Derrick	Dewing Avenue Sooke 2219 Canada
Derry Jeanne	7640 First Ave. Issaquah USA 73980
Spence Michael	Tosca Way Burnaby 337 74674 Canada

o automatickom vyplňaní adries na poštových obálkach môžeme akurát snívať. Napriek vynaloženej námahe sa nám pravdepodobne dosť zásielok vráti ako nedoručené. Ak máme databázu zákazníkov v takomto tvare,

cname	address1	city	postal_code	country
Nowmer Sheri	2433 Bailey Road	Tlaxiaco	15057	Mexico
Whelply Derrick	2219 Dewing Avenue	Sooke	17172	Canada
Derry Jeanne	7640 First Ave.	Issaquah	73980	USA
Spence Michael	337 Tosca Way	Burnaby	74674	Canada

niet čo riešiť, akýkoľvek kancelársky softvér adresy z takto organizovanej databázovej tabuľky, alebo textového súboru načíta a potlač obálke je úplne bezproblémová. Dobré navrhnutý databázový sklad musí umožňovať oveľa zložitejšie dotazy, napríklad nájsť správnu cieľovú skupinu používateľov pre marketingovú kampaň, zistiť či určitý konkrétny zákazník alebo skupina zákazníkov kupuje príslušné produkty, pod pojmom skupina zákazníkov môžeme rozumieť napríklad rodinu, komunitu (teenagerov, seniorov...). Údaje z externých zdrojov majú určitú kvalitu, ktorá je buď postačujúca, alebo nepostačujúca pre ich zavedenie do dátového skladu. Často dokonca býva kvalita údajov značne premenlivá. Preto je potrebné údaje vyčistiť a verifikovať. Jedným z problémov, ktoré musíme pri transformácii údajov riešiť je aj nejednoznačnosť údajov. Napríklad údaj o pohlaví zákazníka môžu byť uložené rôznym spôsobom

name	gender
-----	-----
Nowmer Sheri	Female
Whelply Derrick	male
Derry Jeanne	F
Spence Michael	man

po transformácii by mohol byť tento stĺpec v jednotnom tvare:

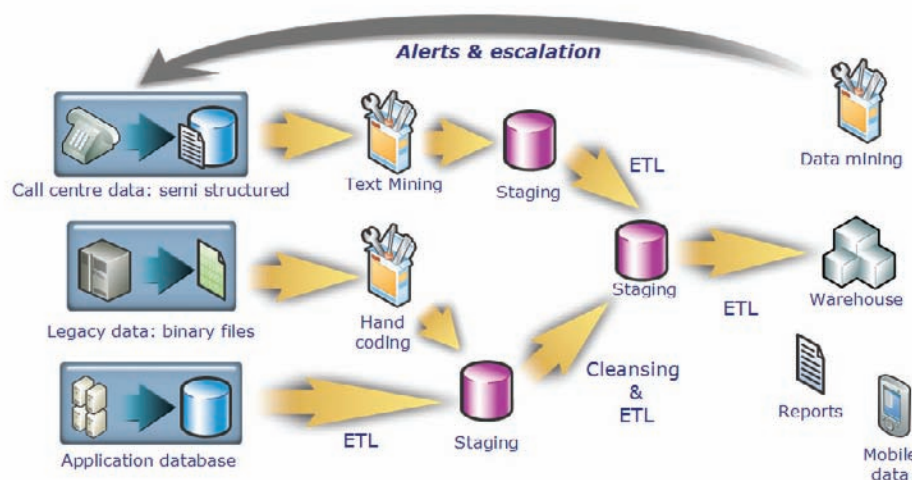
name	gender
-----	-----
Nowmer Sheri	F
Whelply Derrick	M
Derry Jeanne	F
Spence Michael	M

Kapitolou samou o sebe sú na jednej strane chýbajúce údaje a na strane druhej záznamy duplicitné. Duplicita údajov je menším problémom, pretože duplicitné záznamy je možné odstrániť, no v niektorých prípadoch to môže byť pomerne časovo náročné. Väčším problémom sú chýbajúce údaje. Pri malom objeme ich môžeme v niektorých prípadoch ignorovať, alebo doplniť ručne z iných zdrojov. Problémy môžu nastať aj ak sú údaje uložené v rôznych druhoch formátov. Týka sa to rodných čísel, PSČ, čísel účtov a podobne.

Okrem hodnôt samotných sú v údajoch skryté aj rôzne vzťahy, napríklad master – detail, organizačná štruktúra firmy, hierarchická štruktúra zamestnancov a podobne. Údaje sú vo väčšine prípadov dynamické, a v niektorých prípadoch sa neuváženým vymazaním údajov naruší integrita údajov.

## Využitie integračných služieb pre BI projekty

Business Intelligence projekty sa spravidla nezačínajú budovať úplne „nanovo“, teda spôsobom, že doteraz tu nebolo nič a našim cieľom je vybudovať BI systém pre podporu rozhodovania. Spravidla aj v týchto prípadoch používa zákazník pre získavanie informácií pre podporu rozhodovania nejaké provizórne riešenie, napríklad sa generujú výstupné zostavy z transakčných systémov a tieto sú potom buď ručne alebo pomocou automatizačných prvkov softvéru typu Office spracovávané do manažérskych podkladov poskytovaných manažérom pre podporu rozhodovania. Vo všeobecnosti údaje, ktoré chceme aby vstupovali do procesu business intelligence pochádzajú z rôznych nehomogénnych zdrojov. Môžu to byť údaje zo súborových databáz (Access, dBase...), údaje z databáz spravovaných niektorým databázovým serverom (Oracle, Informix, Microsoft SQL Server, Sybase, Interbase, Ingres...), alebo údaje vyexportované nejakou databázovou platformou alebo informačným systémom, napríklad pobočkovou telefónnou ústredňou do tzv. flat file, XML dokumentu a podobne. Príprava a zavedenie údajov je dôležitou súčasťou každého BI riešenia aj riešenie dátového skladu. Údaje z operačného prostredia je potrebné pred zavedením do dátového skladu vyextrahovať, vyčistiť, upraviť a až následne vo vhodnej forme do dátového skladu zaviesť.



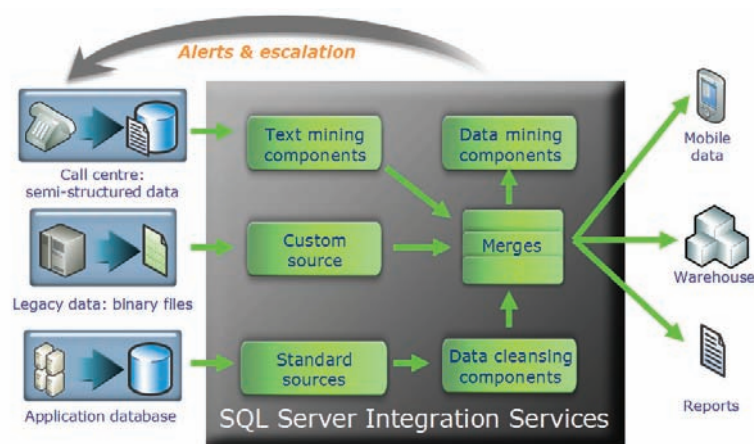
Typický prípad schémy zavádzania údajov z nehomogénnych zdrojov do dátového skladu bez použitia integračných služieb

Predchodcami integračných služieb v predchádzajúcej verzii SQL Servera 2000 boli transformačné služby DTS (Data Transformation Services). Podľa teórie budovania dátových skladov sa tento proces zvykne označovať ETL (Extrahcia, transformácia, Loading) alebo ETT (Extrahcia, Transformácia, Transfer). Hlavnou úlohou tohto procesu je naplnenie dátového skladu určenými údajmi v požadovanom čase..

Jednotlivé etapy procesu ETL sú

- **Extrahcia** — výber dát prostredníctvom rôznych metód
- **Transformácia** — overenie, čistenie, integrovanie a časové označenie dát
- **Loading** — premiestnenie dát do dátového skladu

Aj napriek tomu, že DTS v SQL Serveri 2000 bol jeden z najlepších a najjednoduchšie ovládateľných nástrojov pre jednoduchú konverziu a transformáciu údajov, v novej verzii SQL Servera 2005 boli tieto služby kompletne prepracované a vylepšené, hlavne z hľadiska koncepcie a modelovania zavádzania údajov z nehomogénnych zdrojov do dátového skladu. Inovovaný bol aj názov – Integration Services (integrované služby)



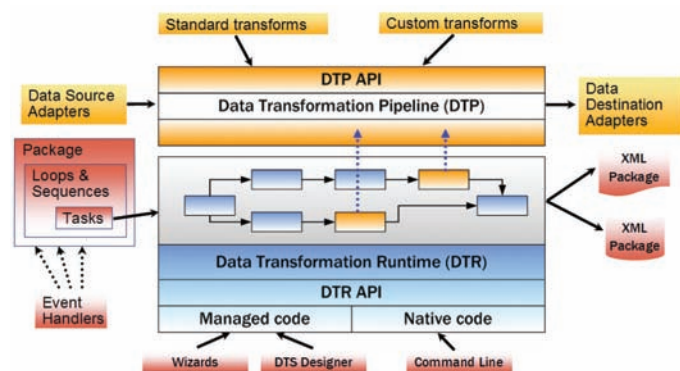
Scenár zavádzania údajov z nehomogénnych zdrojov do dátového skladu s využitím integračných služieb SQL Serveru 2005

Okrem už spomínaného zavádzania údajov z produkčných databáz do dátových skladov nájdú Integrované služby uplatnenie aj pri zbere údajov z rôznorodých systémov a reorganizácii štruktúr údajov, ktoré vyplývajú napríklad zo zmeny organizačnej štruktúry, alebo zo zmeny prípadne reštrukturalizácie predmetu podnikania.

Správne zvládnutá etapa ETL je nevyhnutná v obidvoch popisovaných scenároch, teda aj pre rýchlu a úspešnú migráciu údajov v databázových projektoch aj pre nasadzovanie a prevádzkovanie projektov Business Intelligence a dátových skladoch. V obidvoch prípadoch sa proces ETL zapojí do určitého stavu informačného systému. K dispozícii sú rôzne zostavy, dokumenty a údaje z primárnych transakčných systémov **OLTP** (On-line Transaction Processing).

## Architektúra integračných služieb

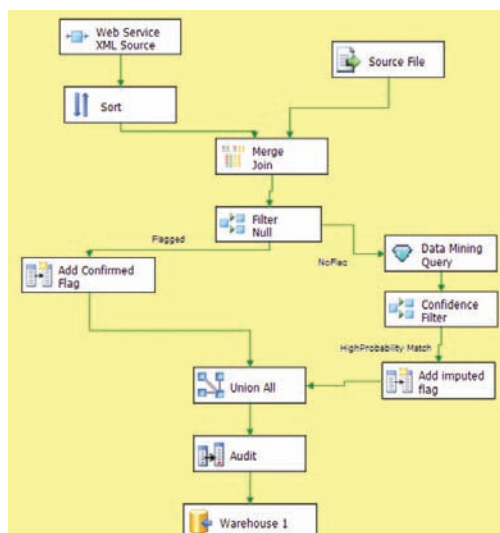
Princíp činnosti Integrovaných služieb je zrejмый z architektonickej schémy. Hlavným blokom architektúry transformačných služieb je Data Transformation Pipeline (DTP), ktorý prepája zdrojové a cieľové dátové adaptéry. Na nižšej vrstve sa vykonávajú komplexné úlohy z ktorých pozostáva proces extrahcie a transformácie.



Architektúra Integrovaných služieb SQL Servera 2005

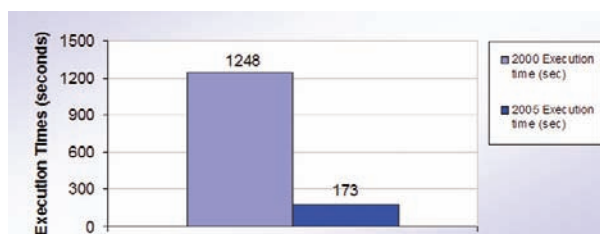


Základným prvkom tejto vrstvy (zreťazené bloky v strede architektonickej schémy) tvoria Tasks, teda samostatne vykonateľné jednoduché čiastkové úlohy podieľajúce sa na komplexnom procese transformácie. Mohli by sme to prirovnať k príkazom v ľubovoľnom programovacom jazyku. Po zreťazení úloh a ich doplnení o rôzne cykly získame komplexnejšie samostatné bloky – balíčky (Packages) Zreťazenie jednotlivých úloh je definované pomocou Task Flow diagramu. Farby čiar so šípkami medzi úlohami znázorňujú v akej podmienkovej vetve sa dané bloky nachádzajú. Napríklad vetva predpokladaného úspešného priebehu je vyznačená zelenou farbou a procesný tok v prípade nespĺnenia nejakej podmienky je znázornená červenou farbou.

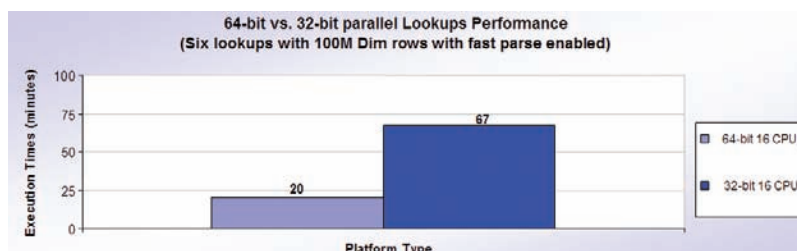


Pomocou Integrovaných služieb je možné vytvárať aj veľmi sofistikované projekty pre spracovanie údajov z heterogénnych zdrojov a ich zavádzanie do dátového skladu

Výrazné vylepšenia Integrovaných služieb vo verzii 2005 umožnili vo väčšine podstatný nárast výkonu prenosu transformácie a zavádzania údajov. Približné porovnanie výkonu Integrovaných služieb SQL Servera 2005 a DTS SQL Servera 2000 je v grafe na obrázku



Porovnanie výkonu Integrovaných služieb SQL Servera 2005 a DTS SQL Servera 2000



Na 64 bitových platformách bežia integračné služby približne 3x rýchlejšie najmä vďaka podpore paralelného spracovania

## Transformácie

Integrované služby SQL Servera 2005 majú implementované pomerne veľké množstvo transformácií, pre pretransformovanie údajov zo zdrojovej do cieľovej formy. Môžeme ich rozdeliť do skupín

### **Business Intelligence**

Transformácie z tejto skupiny využívajú BI funkcionality, napríklad čistenie údajov alebo data miningovú predikciu. Patria sem:

- Fuzzy Grouping Transformation
- Fuzzy Lookup Transformation
- Term Extraction Transformation
- Term Lookup Transformation
- Data Mining Query Transformation

### **Riadkové transformácie**

Riadkové transformácie použijú upravenú hodnotu z niektorého stĺpca záznamu a výsledok transformácie uložia do nového stĺpca.

Môžeme sem zaradiť

- Character Map Transformation
- Copy Column Transformation
- Data Conversion Transformation
- Derived Column Transformation
- Script Component
- OLE DB Command Transformation

### **Rowset Transformácie**

Transformácie z tejto skupiny vytvárajú nové rowsety, ktoré obsahujú agregované a utriedené hodnoty, prípadne hodnoty po operáciách PIVOTa UNPIVOT.

Patria sem:

- Sort Transformation
- Percentage Sampling Transformation
- Row Sampling Transformation
- Pivot Transformation
- Unpivot Transformation

### **Split and Join Transformations**

Tieto transformácie vznikajú spájaním hodnôt z rôznych tabuliek.

Patria sem:

- Conditional Split Transformation
- Multicast Transformation
- Union All Transformation
- Merge Transformation
- Merge Join Transformation
- Lookup Transformation

### **Iné**

Do tejto skupiny patria ostatné preddefinované transformácie, napríklad pre import, zistenie počtu záznamov...

- Export Column Transformation
- Import Column Transformation
- Audit Transformation
- Row Count Transformation
- Slowly Changing Dimension Transformation

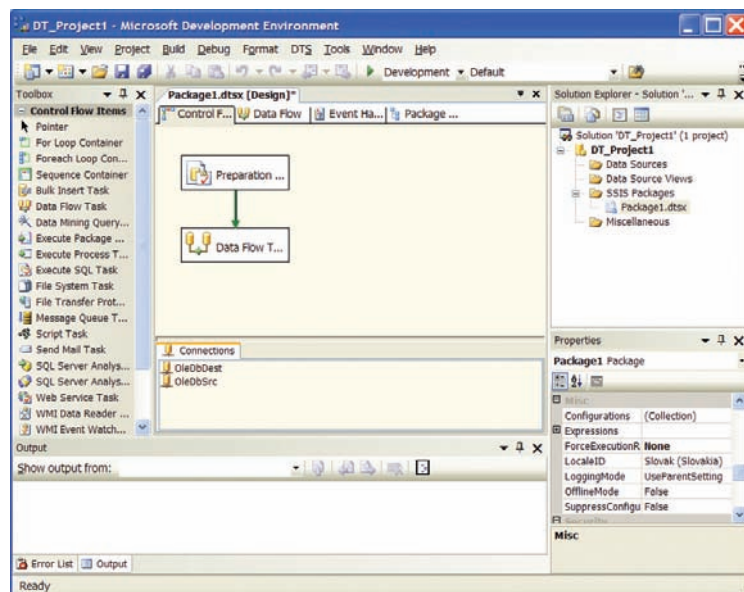
Pre špeciálne prípady je možné definovať vlastnú transformáciu.

## Integračné služby v prostredí vývojového prostredia Visual Studio 2005

Najskôr sa zoznámime s možnosťami a usporiadaním obrazovky vývojového prostredia v režime Integration Services Project.

Pre vizuálne modelovanie, návrh, vytváranie, testovanie a ladenie projektov pre prevod transformáciú a integráciu údajov slúži nástroj Designer, ktorý je súčasťou integrovaného prostredia Business Intelligence Development Studio. Designer obsahuje dva oddelené editory, jeden pre návrh dátových tokov a jeden pre návrh riadiacich tokov. Tomu je prispôsobená aj pracovná plocha vývojového prostredia. Jej hlavné okno je rozdelené na štyri záložky

- Control Flow
- Data Flow
- Event Handlers
- Package Explorer

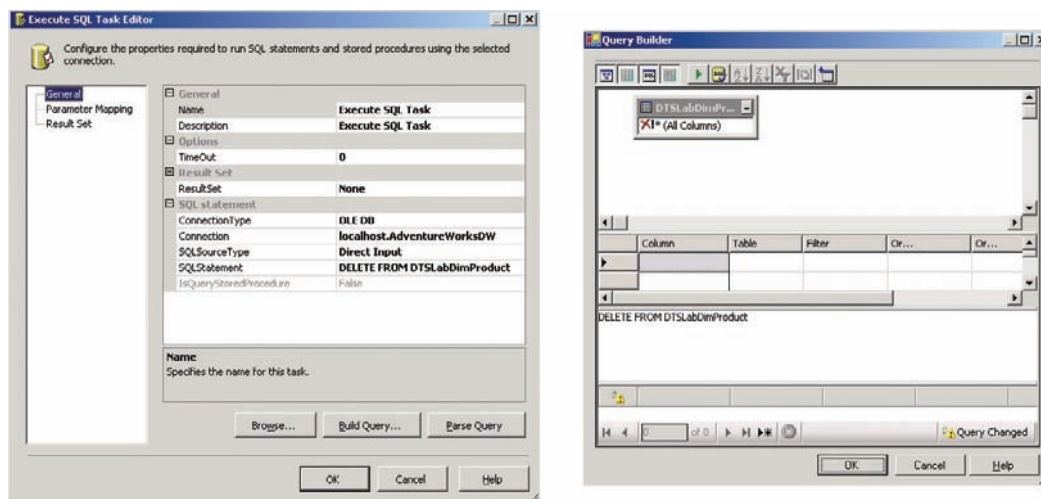


BI DevStudio – záložka ControlFlow

Približné poradie úkonov pri návrhu integračného projektu nám naznačujú zložky v okne Solution Explorer. Filozofia usporiadania a návaznosti zložiek v tomto okne vyplýva z koncepcie UDM modelovania (viď predchádzajúcu kapitolu). Okno obsahuje štyri zložky

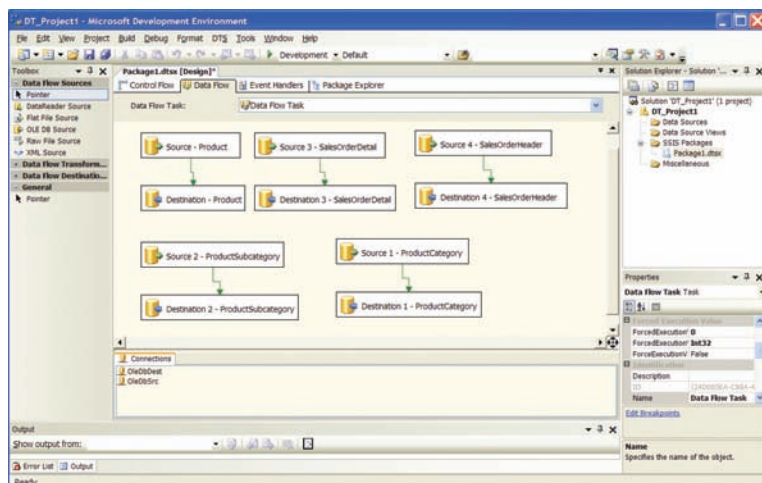
- Data Sources
- Data Source Views
- SSIS Packages
- Miscellaneous

V režime návrhu jednoduchého SSIS projektu sú v záložke Control Flow umiestnené ikony jednotlivých čiastkových úloh, z ktorých sa projekt skladá. Pre návrh úloh, ktoré sa vykonávajú pomocou kódu v jazyku SQL slúži Execute SQL Task Editor. Aktivujeme ho v kontextovom menu pomocou položky Execute Task. V dialógovom okne môžeme nastavovať rôzne parametre, prípadne vytvárať SQL skripty pomocou nástroja Query Builder.



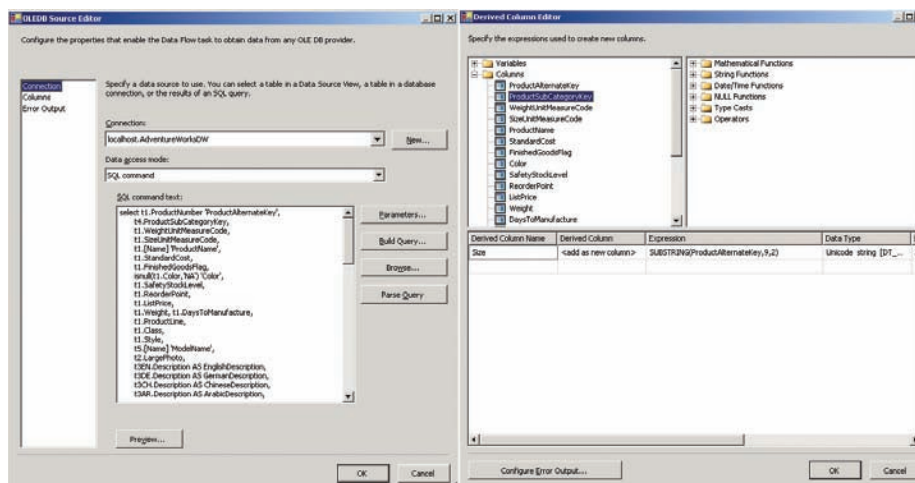
Execute SQL Task Editor a Query Builder

Dvojklikom na symbol Data Flow Task sa prepne zo záložky Control Flow do záložky Data Flow. Zobrazí sa diagram dátových tokov. Môže sa jednať o zložitú konverziu a zlučovanie údajov do jednej tabuľky, prípadne podobne ako znázorňuje diagram na obrázku môžeme len prenášať viacero databázových tabuliek zo zdrojovej do cieľovej databázy bez výraznejších úprav.



BI DevStudio – záložka Data Flow

V dolnej časti obrazovky vývojového prostredia je okno so symbolmi pripojení na dátové zdroje.



OLEDB Source Editor a a Derived Column Editor

Po spustení SSIS projektu môžeme v okne Control Flow sledovať priebeh jeho vykonávania. Stav vzhľadom na jednotlivé objekty je signalizovaný farbou ich symbolov. Význam jednotlivých farieb je nasledovný

- Šedá – čakanie na spustenie
- Žltá – aktuálne spustené
- Zelená – úspech
- Červená – chyba

### Predstavenie databázy AdventureWorks

Aby sme sa v cvičných aplikáciách pre BI, obzvlášť v oblasti integračných služieb čo najviac priblížili realite, teda zavedenia údajov z relačných OLTP databáz do dátových skladov, využijeme cvičnú OLTP databázu AdventureWorks, ktorá je dodávaná spolu s SQL Serverom 2005. Aby sme mohli s touto cvičnou databázou efektívne pracovať, je potrebné sa s ňou aspoň letmo zoznámiť.

Nakoľko SQL Server 2005 umožňuje zoskupovať objekty do schém, aj databáza AdventureWorks je takto prehľadne organizovaná

Schema	Popis objektov	Tabulky
HumanResources	Zamestnanci spoločnosti Adventure Works Cycles.	Employee Department
Person	Mená a adresy zákazníkov, predajcova zamestnancov.	Contact Address StateProvince
Production	Produkty vyrábané a predávané spoločnosťou Adventure Works Cycles.	BillOfMaterials Product WorkOrder
Purchasing	Dodávatelia súčiastok.	PurchaseOrderDetail PurchaseOrderHeader Vendor
Sales	Údaje týkajúce sa obchodu a zákazníkov	Customer SalesOrderDetail SalesOrderHeader

Pre čitateľov, ktorí boli zvyknutí používať v predchádzajúcej verzii SQL Servera 2000 jeho cvičné databázu NORTHWIND uvádzame zaujímavú približnú analógiu ktorým tabuľkám z tejto databázy fiktívnej firmy sa približne svojou štruktúrou a použitím podobajú tabuľky novej databázy

Northwind	AdventureWorks
Categories	Production.ProductCategory
Customers	Sales.Customer Join with Sales.Individual and Sales.Store
Customer Demographics	Sales.Individual Sales.Store
Employees	HumanResources.Employee Join with Person.Contact
Employee Territories	Sales.SalesPerson
Orders	Sales.SalesOrderHeader
Order Details	Sales.SalesOrderDetail
Products	Production.Product
Region	Sales.SalesTerritory
Shippers	Purchasing.ShipMethod
Suppliers	Purchasing.Vendor
Territories	Sales.SalesTerritory

### Úvodný príklad pre integrované služby

Predmetom prvého cvičného príkladu nad databázou AdventureWorks bude vytvorenie operačného dátového tržiska nad údajmi z OLTP databázy. Technicky zjednodušene povedané potrebujeme preniesť niekoľko tabuliek zo zdrojovej do cieľovej databázy. Pre tento účel môžeme z databázy AdventureWorks využiť napríklad tabuľky zo schém SALES, kde sú uložené údaje z oblasti marketingu a predaja firmy AdventureWorks.

Spoločnosť má dva typy zákazníkov, ktorí sa odlišujú pomocou atribútu CustomerType .

- individuálnych, ktorí nakupujú cez webový on-line obchod (CustomerType = ,I')
- obchody s bicyklami a cyklistickými potrebami (CustomerType = ,S')

Nakoľko firemné procesy nie sú izolované ale navzájom previazané, potrebujeme pre niektoré výpisy týkajúce sa predaja a marketingu aj tabuľky z iných schém. Nakoľko do všetkých procesov vstupujú osoby (zamestnanci, obchodníci, zákazníci), ich údaje sú uložené v tabuľkách schémy Person. Pre náš príklad využijeme tabuľky objednávok

- Sales.SalesOrderHeader
- Sales.SalesOrderDetail

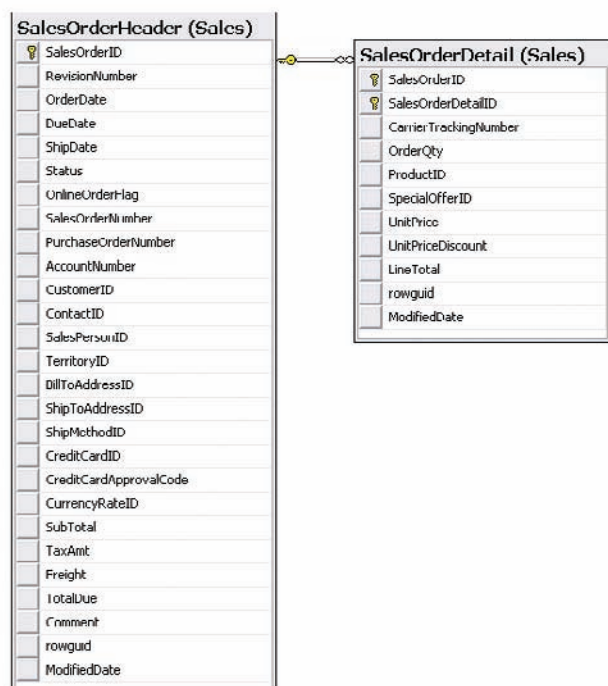


Diagram väzieb tabuliek pre výpis údajov o objednávkach AdventureWorks použitých v príklade

Druhá schéma, ktorú v príklade využijeme bude PRODUCTION. Firma AdventureWorks vyrába bicykle a rôzne cyklistické príslušenstvo. Údaje ohľadne produktov firmy sú uložené v tabuľkách schémy Production.. V príklade využijeme z tejto schémy tabuľky:

- Production.Product
- Production.ProductSubcategory
- Production.ProductCategory

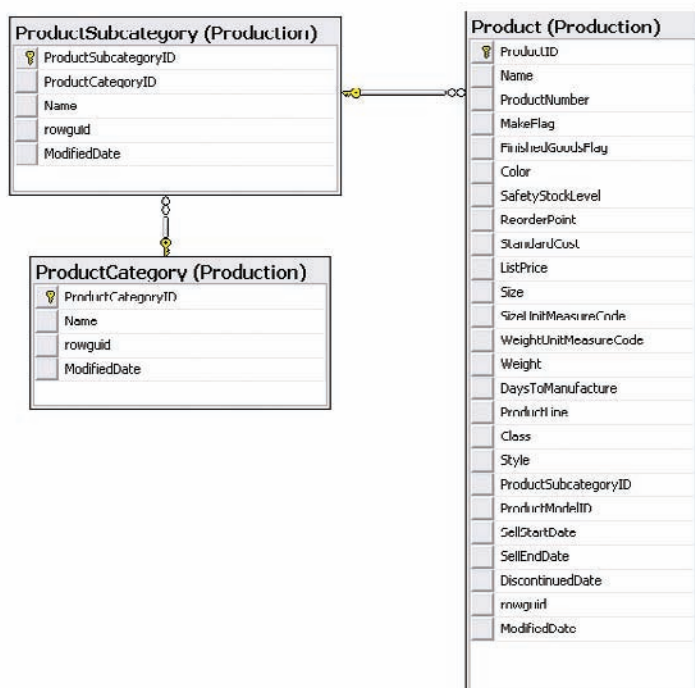
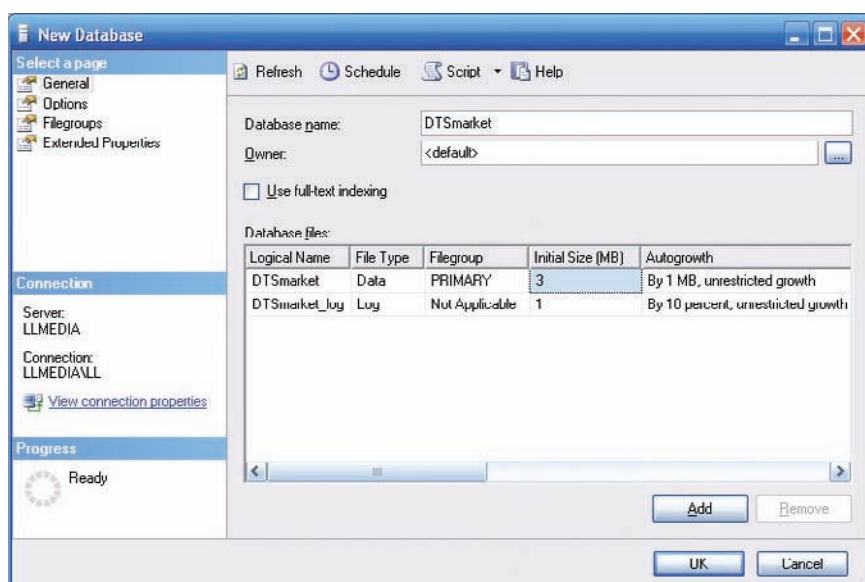


Diagram väzieb tabuliek pre výpis produkcie firmy AdventureWorks použitých v príklade

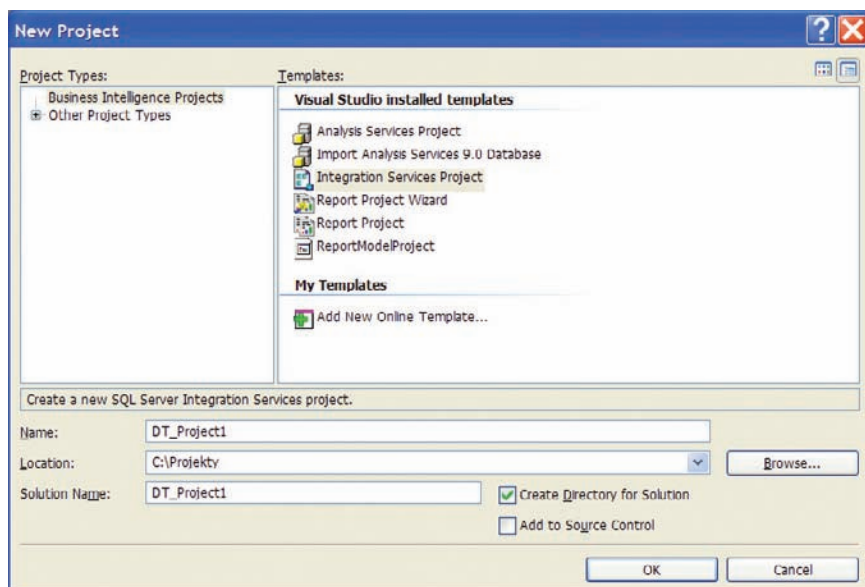
Schémy použitých tabuliek sme nepublikovali náhodou. Všimnite si jednu z teoretického hľadiska dosť podstatnú záležitosť. Chýba tu tabuľka, ktorá by bola jednoznačným podkladom pre časovú dimenziu. Z vybraných tabuliek však časové údaje obsahuje tabuľka OrderHeader. Takto to býva aj v praxi. Každý si ukladá do databáz údaje o predmete svojho podnikania a tieto samozrejme obsahujú aj dátumové a časové údaje.

Ako prvý prípravný krok vytvoríme cieľovú databázu napríklad s názvom DTSMarket. Postup vytvorenia novej databázy v SQL Server Management Studiu je triviálny: v okne Objekt Explorer aktivujeme v zložke Databases kontextové menu Create New Database.

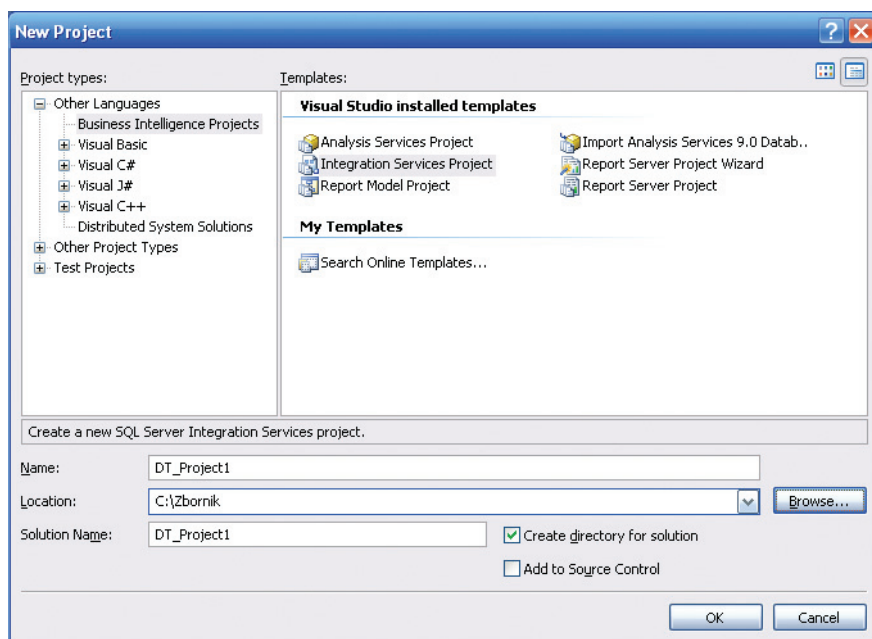


Dialog pre vytvorenie novej databázy v SQL Server Management Studiu

Ďalšie kroky budeme vykonávať v prostredí Business Intelligence Development Studio.



*Integration Services Project – BI Dev Studio*



*Integration Services Project – vytvorenie nového IS projektu v plnej verzii Visual Studia 2005*

Pomocou menu *Start | Programs | Microsoft SQL Server | Business Intelligence Development Studio* spustíme integrované vývojové prostredia a v ňom vytvoríme nový projekt (*File | New | Project*) typu Information Services Project. Nájde ho v záložke Business Intelligence Projects. Nazveme ho napríklad DT\_Project1. V zložke SSIS Packages bude automaticky vytvorený balíček Package.dtsx. V tomto prvom príklade však tento balíček potrebovať nebudeme, takže ho môžeme vymazať. Použijeme kontextové menu *SSIS Import and Export Wizard* zložky *SSIS Packages*. Aktivujeme tým rovnako pomenovaného sprievodcu.





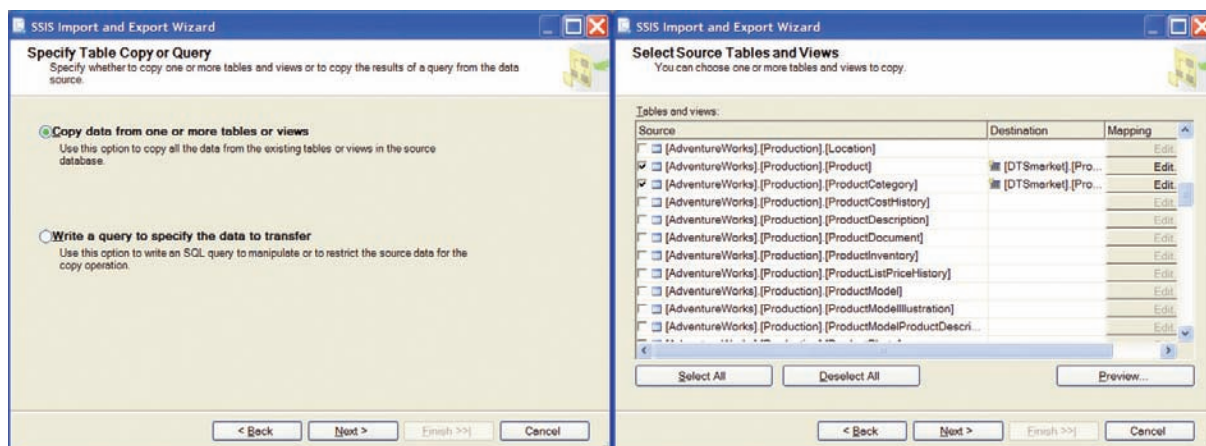
SSIS Import and Export Wizard – výber zdrojovej databázy

Podľa pokynov sprievodcu najskôr vyšpecifikujeme zdrojovú databázu. Vyberieme druh pripojenia, a špecifikujeme názov servera a názov zdrojovej databázy, v našom prípade AdventureWorks. Ako cieľovú databázu vyberieme zatiaľ prázdnu nami vytvorenú databázu DTSMarket.

V nasledujúcom kroku máme možnosť zvoliť si metódu pre výber množiny údajov, ktoré potrebujeme preniesť a prípadne pretransformovať. Máme na výber dve možnosti:

- kopírovať vybrané tabuľky a pohľady ako celky zo zdroja dát do cieľovej databázy
- pomocou Query dotazu špecifikovať podmnožiny údajov, ktoré potrebujeme pretransformovať.

V kľúčovom dialógu sprievodcu vyberieme tabuľky, ktoré chceme preniesť zo zdrojovej do cieľovej databázy. Zatiaľ v tomto jednoduchom príklade nebudeme nič upravovať ani transformovať.



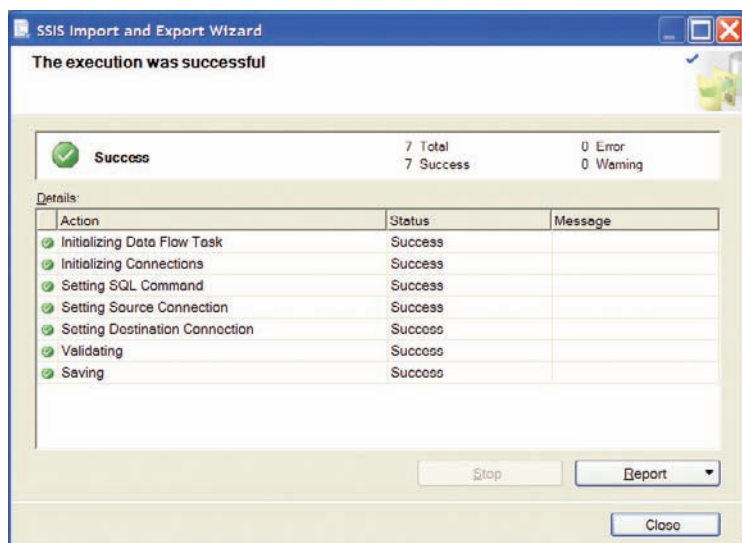
Výber metódy importu a výber tabuliek určených pre prenos do novej databázy

V dialógu označíme tabuľky

- Sales.SalesOrderHeader
- Sales.SalesOrderDetail
- Production.Product
- Production.ProductSubcategory
- Production.ProductCategory

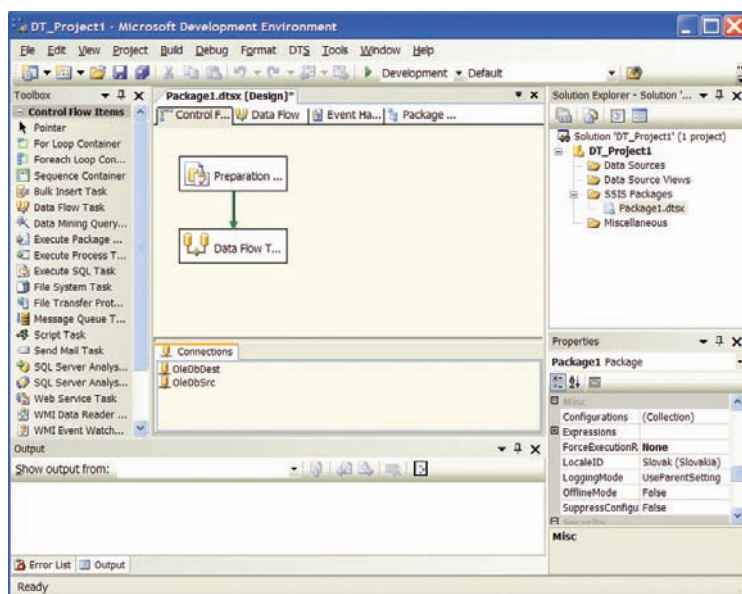
Alternatívne by sme nemuseli prenášať údaje po celých tabuľkách, ale využiť možnosť vytvorenia SQL dotazu pre výber podmnožiny údajov zo zdrojovej databázy do cieľovej. Finálnym krokom sprievodcu je vytvorenie SSIS tasku. Priebeh tohto procesu, teda vynášania metadát môžeme sledovať v dialógu Performing Operation. Tu si treba uvedomiť, že v tejto fáze sa ešte s údajmi samotnými zatiaľ nepracuje. Mierne

zavádzajúce je, že podobný dialóg v predchádzajúcej verzii SQL Servera 2000 znázorňoval priebeh samotnej transformácie



Priebeh zostavovania tasku pre prenos údajov zo zdrojovej do cieľovej databázy

Výsledok zostavovania tasku je zobrazený na pracovnej obrazovke vývojového prostredia v záložkách Control Flow a DataFlow.



BI Development Studio – Control Flow

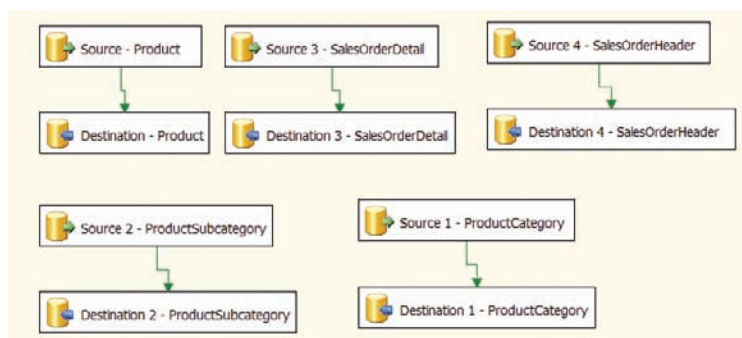
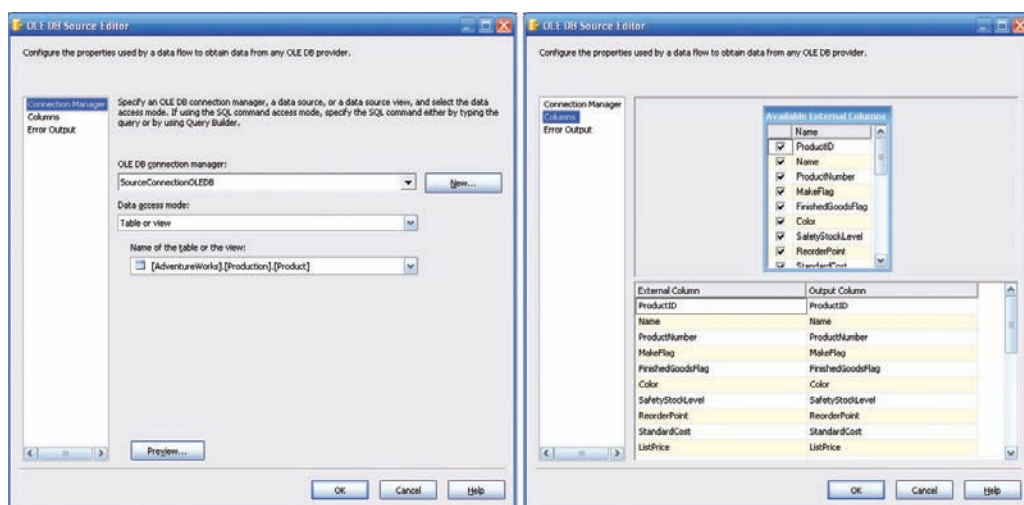


Diagram v záložke Data Flow

V záložke Data Flow vidíme nadefinovaných päť vzájomne nesúvisiacich úloh. Pre každú prevádzanú tabuľku je znázornený tok údajov zo zdrojového do cieľového úložiska. Tieto úložiská sú znázornené obdĺžnikmi. Veľmi dôležitým symbolom je aj šípka medzi nimi, kde sú nadefinované transformačné operácie. Kliknutím na túto šípku aktivujeme Data Flow Path editor, kde si môžeme prehliadnuť metadata. Napríklad pre tabuľku Product Category sú v tvare

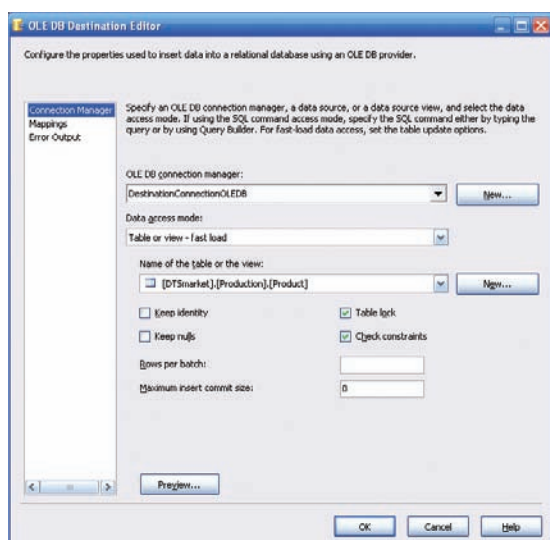
```
,Name``Data Type``Precision``Length``Source Component`
,ProductCategoryID``DT_I4``0``0``0``0``Source 1 - ProductCategory`
,Name``DT_WSTR``0``0``50``0``Source 1 - ProductCategory`
,rowguid``DT_GUID``0``0``0``0``Source 1 - ProductCategory`
,ModifiedDate``DT_DBTIMESTAMP``0``0``0``0``Source 1 - ProductCategory`
```

Kliknutím na symbol zdroja alebo cieľa údajov vyvoláme Source Editor, prípadne Destination Editor



Source editor, vľavo záložka Connection Manager, vpravo záložka Columns

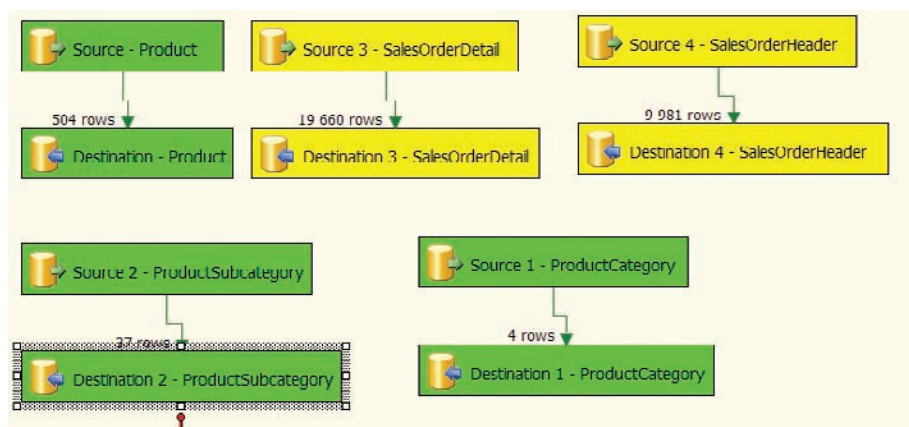
V tejto etape môžeme napríklad pomocou Source Editora v záložke Columns vynechať z transformácie stĺpce, ktoré v cieľových tabuľkách nepotrebujeme



OLE DB Destination Editor

Samotnú úlohu spustíme zelenou šípkou na toolbare vývojového prostredia. Na procesnom diagrame môžeme podľa farieb jednotlivých symbolov sledovať priebeh vykonávania úlohy, pričom aktuálne bežiacie úlohy sú znázornené žltým pozadím symbolu, úspešne ukončené úlohy zeleným pozadím a prípadne chybné ukončené

úlohy červeným pozadím symbolu. Zároveň môžeme podľa čísel nad blokmi cieľovej destinácie informatívne sledovať počet spracovaných a prenesených záznamov



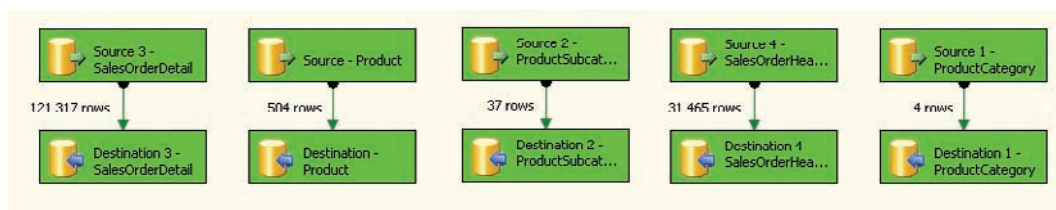
Priebeh vykonávania SSIS tasku je signalizovaný farbou pozadia jednotlivých blokov

Ak nám diagram po dobehnutí všetkých úloh „ozelenie“, transformácia údajov prebehla úspešne. V prípade problémov nám pomôže napríklad protokol o priebehu SSIS tasku, ktorý sa zobrazuje v okne Output v dolnej časti obrazovky vývojového prostredia

```

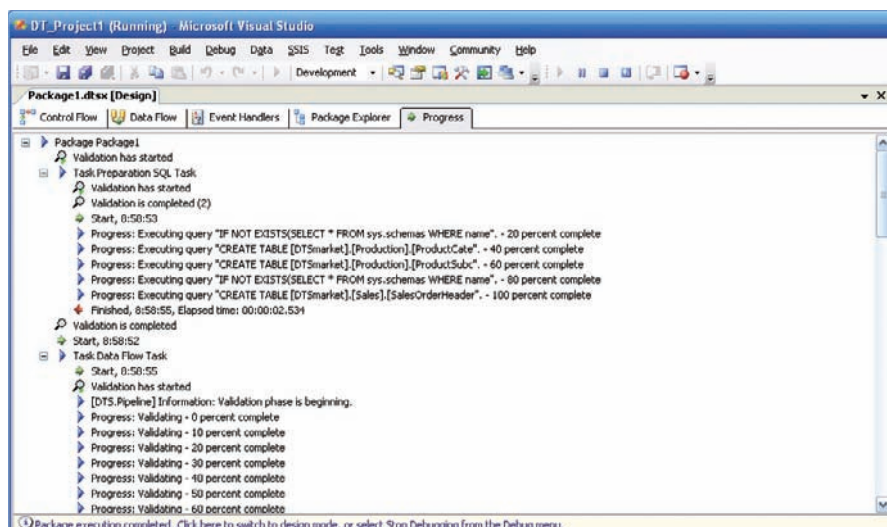
SSIS package „Packagel.dtsx“ starting.
DTS.Pipeline: Validation phase is beginning.
DTS.Pipeline: Prepare for Execute phase is beginning.
DTS.Pipeline: Pre-Execute phase is beginning.
DTS.Pipeline: Execute phase is beginning.
Destination 2 - ProductSubcategory [237]: The final commit for the data insertion has
started.
Destination 1 - ProductCategory [183]: The final commit for the data insertion has
started.
Destination - Product [90]: The final commit for the data insertion has started.
Destination - Product [90]: The final commit for the data insertion has ended.
Destination 1 - ProductCategory [183]: The final commit for the data insertion has ended.
Destination 2 - ProductSubcategory [237]: The final commit for the data insertion has
ended.
Destination 4 - SalesOrderHeader [445]: The final commit for the data insertion has
started.
Destination 4 - SalesOrderHeader [445]: The final commit for the data insertion has
ended.
Destination 3 - SalesOrderDetail [311]: The final commit for the data insertion has
started.
Destination 3 - SalesOrderDetail [311]: The final commit for the data insertion has
ended.
DTS.Pipeline: Post Execute phase is beginning.
DTS.Pipeline: Cleanup phase is beginning.
DTS.Pipeline: „component „Destination - Product“ (90)“ wrote 504 rows.
DTS.Pipeline: „component „Destination 1 - ProductCategory“ (183)“ wrote 4 rows.
DTS.Pipeline: „component „Destination 2 - ProductSubcategory“ (237)“ wrote 37 rows.
DTS.Pipeline: „component „Destination 3 - SalesOrderDetail“ (311)“ wrote 121317 rows.
DTS.Pipeline: „component „Destination 4 - SalesOrderHeader“ (445)“ wrote 31465 rows.
SSIS package „Packagel.dtsx“ finished: Success.

```



Ak SSIS task prebehne úspešne, všetky symboly Data Flow diagramu sú zelené a vedľa symbolov transformácie (čiara so šípkou) je vypísaný počet prenesených údajov

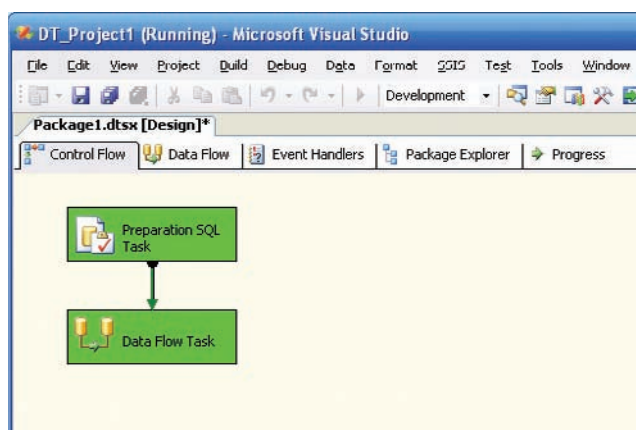
Ak spúšťame alebo ladíme zložitý projekt a navyše vo veľkej databáze, môže beh SSIS tasku trvať aj pomerne dlho. Priebeh tasku môžeme sledovať v záložke Progress



Priebeh SSIS tasku môžeme sledovať v záložke Progress

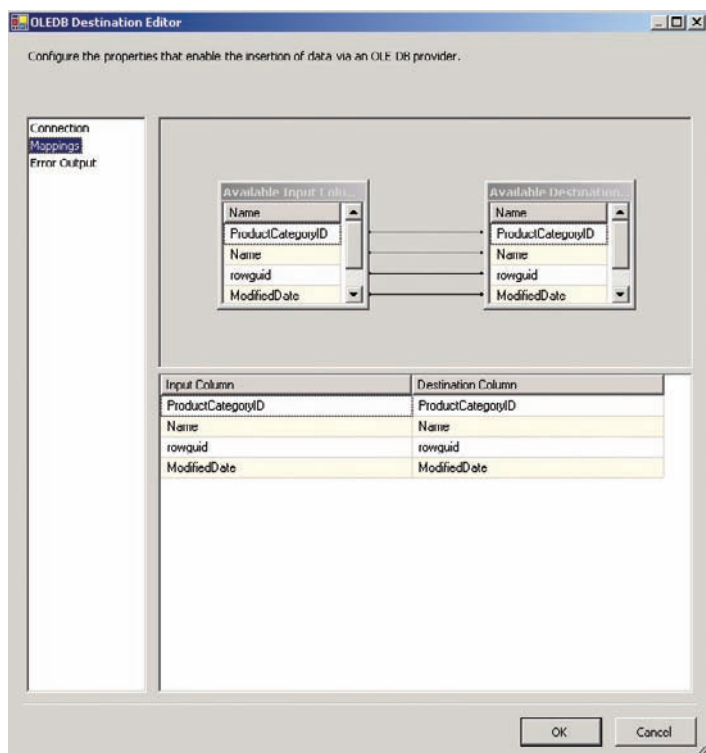
V záložke Control Flow sa zobrazuje priebeh spustenia tasku po jednotlivých funkčných blokoch. V našom jednoducho príklade kopírovania údajov zo zdrojových tabuliek do cieľovej databázy má tento diagram len dva bloky

- Preparation SQL Task
- Data Flow Task



Úspešné ukončenie SSIS tasku

Pre neveriacich Tomášov je tu SQL Server Management Studio, kde si môžu v databáze DTSmarket pozrieť zoznam a obsah novovytvorených tabuliek.



Prehľad tabuliek v novovytvorenej databáze

... úloha pokračuje úvodným príkladom v nasledujúcej kapitole OLAP analýzou údajov z dátového tržiska..

## Kapitola 4: OLAP analýza údajov v databázach

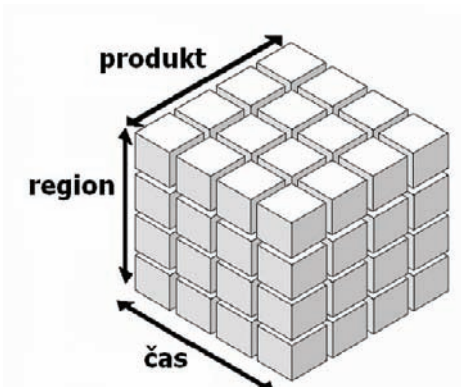
*K tejto kapitole je možné pristupovať dvomi spôsobmi, buď si najskôr preštudovať (alebo zopakovať) teoretický úvod do problematiky OLAP, alebo si najskôr pozrieť príklad (oveľa lepšie než pozrieť je skúsiť) k teoretickému úvodu sa vrátiť neskôr.*

### Teoretický úvod

Základom pre vytvorenie OLAP kocky sú fakty a dimenzie. Ako sa dozvieme neskôr tabuľky faktov a dimenzií môžu vytvárať určité schémy, napríklad hviezdicovú schému (star schema) alebo schému snehovej vločky (snowflake schema).

**Fakty** sú numerické merné jednotky obchodovania. Tabuľka faktov je spravidla najväčšia tabuľka v databáze a obsahuje veľký objem dát. Niektoré jednoduchšie databázy zvyčajne obsahujú len jednu tabuľku faktov, iné, hlavne DSS schémy môžu obsahovať viaceré tabuľky faktov. Prvotné fakty, napríklad objem predaja, sa môžu kombinovať alebo vypočítať pomocou iných faktov vytvoriť tak merné jednotky. Merné jednotky sa môžu uložiť v tabuľke faktov, prípadne vyvolať, ak je to nevyhnutné, na účely vykazovania.

**Dimenzie** obsahujú logicky alebo organizačne hierarchicky usporiadané údaje. Sú to vlastne textové popisy obchodovania. Tabuľky dimenzií sú zvyčajne menšie ako tabuľky faktov a dáta v nich sa nemenia tak často. Tabuľky dimenzií vysvetľujú všetky „prečo“ a „ako“ pokiaľ ide o obchodovanie a transakcie prvkov. Kým dimenzie vo všeobecnosti obsahujú relatívne stabilné dáta, dimenzie zákazníkov sa aktualizujú častejšie. Veľmi často sa používajú časové, produktové a geografické dimenzie. Narastajúcim počtom dimenzií geometrickým radom narastá veľkosť (množstvo údajov) OLAP kocky.



Príklad dimenzií

Tabuľky dimenzií obvykle obsahujú stromovú štruktúru. Napríklad dimenzia vytvorená na základe geografických informácií, teda regionálna dimenzia sa člení na jednotlivé úrovne podľa konkrétneho územnosprávneho členenia danej geografickej oblasti, napríklad (počet bodiek znamená úroveň vnorenia jednotlivých atribútov dimenzie):

#### Region

- Kontinent,
- ● Krajina,
- ● ● Územný celok,
- ● ● ● Mesto

Hierarchicky býva usporiadaná aj produktová klasifikácia. Produkty sa začleňujú ho druhov, kategórií, skupín a podobne, napríklad:

#### Produkt

- Druh produktu,
- ● Kategória,
- ● ● Subkategória,
- ● ● ● Názov produktu

Takmer vždy sa používa ako dimenzia čas, kde sú jednotlivé úrovne definované podľa kalendárnych alebo fiškálnych zvyklostí napríklad:

Čas

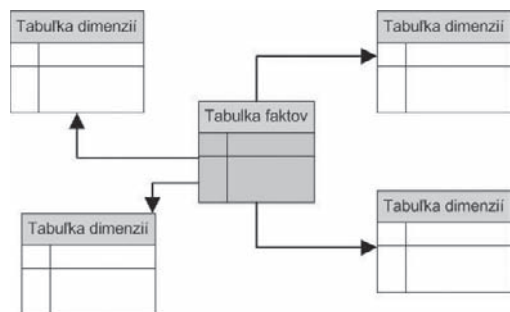
- Rok,
- • Kvartál,
- • • Mesiac,
- • • • Týždeň

Ak uvedené údaje usporiadame do vhodne logicky organizovanej multidimenzionálnej štruktúry, budeme pracovať s oveľa menšími rozmermi dimenzií a údaje budú mať oveľa väčšiu vypovedaciu schopnosť. Multidimenzionálna informácia je navyše v čistej podobe, to znamená, že je orientovaná na predmet podnikania a nie je viazaná na konkrétny systém, ktorý je určený na zber a spracovanie základných údajov.

### Schémy tabuliek faktova dimenzií

Multidimenzionálne kocku (príklad vidíme na obrázku 1) vytvárame na základe dimenzionálneho modelu, ktorý má určité topologické usporiadanie, ktorému hovoríme schéma. Najčastejšie používame hviezdicovú schému (star schema), alebo schému „snehovej vločky“ (snowflake schema)

**Hviezdicová schéma** sa skladá z tabuľky faktov obsahujúcej cudzie kľúče, ktoré sa vzťahujú k primárnym kľúčom v tabuľkách dimenzií. Hviezdicová schéma nemá normalizované dimenzie ani relačné prepojenie medzi tabuľkami dimenzií, preto je veľmi ľahko pochopiteľná, ale v dôsledku nenormalizovaných dimenzií je vytvorenie takéhoto modelu relatívne pomalé, ale na druhej strane, tento model poskytuje vysoký „dopytovací výkon“



Hviezdicová schéma (star schema)

**Schéma „snehovej vločky“** obsahuje niektoré dimenzie zložené z viacerých relačne zviazaných tabuliek. Tento model umožňuje rýchlejšie zavedenie údajov do normalizovaných tabuliek, ale má podstatne nižší dopytovací výkon, lebo obsahuje väčšie množstvo spojení tabuliek

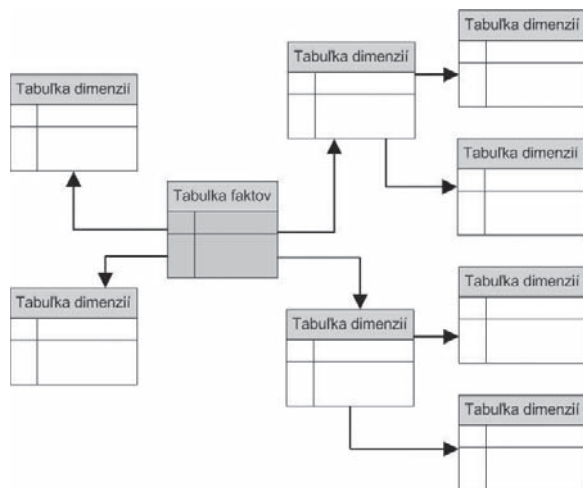


Schéma snehovej vločky (snowflake schema)



## Porovnanie relačných a multidimenzionálnych databáz

Obidva typy databáz majú svoje výhody a nevýhody z ktorých potom vyplýva scenár nasadenia a použitia.

**Relačná databáza** Údaje sú uložené v dvojdimenzionálnych tabuľkách. Každý riadok v tabuľke obsahuje dáta, ktoré sú spravidla obrazom reálneho sveta, teda dáta, ktoré sa vzťahujú k nejakej veci alebo k jej časti. Stĺpce dvojdimenzionálnych databázových tabuliek obsahujú údaje týkajúce sa atribútu.

**Multidimenzionálna databáza.** Zjednodušene povedané, OLAP kocka je vlastne ekvivalent tabuľky v relačnej databáze. Z hľadiska programátorského je to určitý typ viacrozmerného poľa. Každá kocka má niekoľko dimenzií (ekvivalent indexových poli v relačných tabuľkách). Azda najlepšie si dokážeme predstaviť klasickú trojrozmernú kocku, no počet dimenzií v reálnych multidimenzionálnych databázach je spravidla väčší. Priestor pre celú kocku je už vopred rozvrhnutý. Jednotlivé záznamy sa v multidimenzionálnych kockách nachádzajú na priesečníkoch dimenzií. S rastúcim počtom rozmerov multidimenzionálnej databázy veľmi rýchlo rastú aj požiadavky na úložnú kapacitu. Nie na všetkých priesečníkoch dimenzií sa vždy nachádzajú údaje. Takúto kocku nazývame aj riedkou kockou. V praxi sa u multidimenzionálnych databáz používajú rôzne technológie na kompresiu objemu použitého diskového priestoru.

## Spôsob ukladania multidimenzionálnych údajov

### Multidimenzionálny OLAP (MOLAP)

Pre multidimenzionálne on-line analytické spracovanie (MOLAP) sa získavajú dáta buď z dátového skladu alebo z operačných zdrojov. Mechanizmus MOLAP potom uloží analytické dáta vo vlastných dátových štruktúrach a sumároch. Počas tohoto procesu sa napočíta toľko predbežných výsledkov, koľko je z technického a časového hľadiska možné. Údaje v úložisku typu MOLAP sa teda budú ukladať ako vopred vypočítané pole. Hodnoty dát aj indexov sa uchovávajú v jednotlivých poliach multidimenzionálnej databázy. Databáza je organizovaná tak, aby umožnila rýchle získavanie príslušných údajov z viacerých dimenzií. Časť údajov sa môže zaviesť zo servera ku klientovi, čo umožňuje rýchle analýzy bez veľkého zaťaženia siete. Hlavnou výhodou MOLAP je maximálny výkon vzhľadom na dopyty používateľa, nevýhodou je redundancia údajov, nakoľko tieto sú uložené jednak v relačnej databáze, jednak v multidimenzionálnej databáze. Požiadavky na úložnú kapacitu môžu v prípade použitia viacerých dimenzií extrémne narastať.

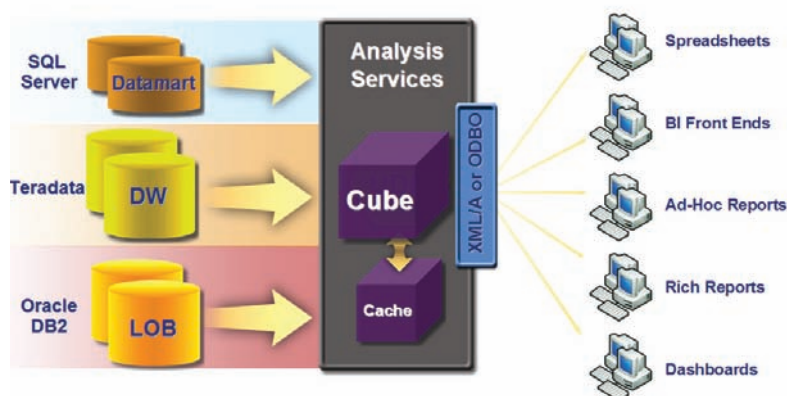
### Relačný databázový OLAP (ROLAP)

Relačné on-line analytické spracovanie údajov (ROLAP) získava údaje pre analýzy z relačného dátového skladu. Tieto údaje z relačných databáz sa po spracovaní predkladajú používateľovi ako multidimenzionálny pohľad. Dáta a metadáta sa pri úložisku ROLAP ukladajú ako záznamy v relačnej databáze. OLAP server dynamicky používa tieto metadáta na generovanie SQL príkazov, ktoré sú potrebné na získavanie dát požadovaných používateľom. Pri tomto spôsobe zostávajú dáta uložené v relačných databázach, takže nevzniká problém z redundanciou.

### Hybridný OLAP (HOLAP)

Hybridné OLAP je kombináciou úložisk MOLAP a ROLAP, pričom sa využívajú výhody jednotlivých typov úložisk a do značnej miery sa eliminujú nevýhody. Údaje zostávajú v relačných databázach a napočítané agregácie sa ukladajú do multidimenzionálnych štruktúr. Pri dopytovaní sa údaje vyberajú do multidimenzionálnej pamäti cache. V hybridnom riešení relačná databáza ukladá množstvo detailných dáta multidimenzionálny model ukladá sumárne dáta.

*Na úvod praktickej časti kapitoly venovanej vytváraniu OLAP kociek apelujeme na čitateľa, ktorý doteraz nečítal tretiu kapitolu venovanú UDM (Unified Dimension Model), aby sa najsôr vrátil k tejto niekoľkostranovej kapitole, ktorá popisuje základné princípy a filozofiu vytvárania BI štruktúr. Na základe princípov UDM sú totiž koncipované všetky čiastkové úkony, ktoré vedú k vytvoreniu OLAP kociek.*



Komplexná schéma architektúry analytických služieb SQL Servera 2005

## Úvodný príklad pre vytvorenie OLAP kocky

... pokračovanie úvodného príkladu z predchádzajúcej kapitoly...

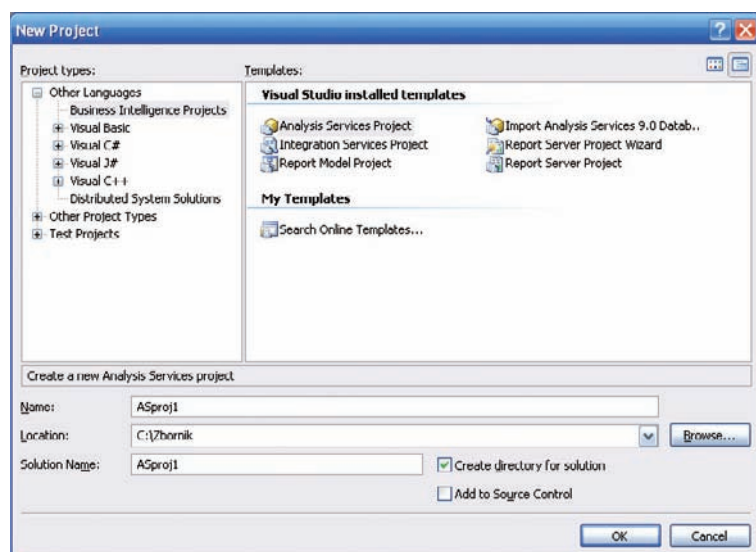
V predchádzajúcej kapitole sme vytvorili jednoduché dátové tržisko (samostatnú databázu) DTMarket, do ktorej boli z databázy AdventureWorks presunuté tabuľky

- Sales.SalesOrderHeader
- Sales.SalesOrderDetail
- Production.Product
- Production.ProductSubcategory
- Production.ProductCategory

Tabuľky boli zo zdrojovej databázy do cieľovej presunuté bez akýchkoľvek úprav transformácií, preto tento príklad môžu pokojne urobiť aj čitatelia, ktorí príklad z predchádzajúcej kapitoly vynechali. Vtedy pracujeme s originálnou zdrojovou databázou Adventure Works

V tejto kapitole nadviažeme vytvorením Universal Dimensional Modelu (UDM) alebo OLAP kocky

V o vývojovom prostredí Business Intelligence Development Studio vytvoríme nový projekt typu Analysis Services Project napríklad s názvom Asproj1.



Nový projekt typu Analysis Services Project

Postup v akom budeme novú aplikáciu vytvárať (vlastne modelovať) je naznačený poradím zložiek v okne vývojového prostredia Solution Explorer v pravom hornom rohu. OLAP analýzy sa týkajú zložky

- Data Sources
- Data Source Views
- Cubes
- Dimension

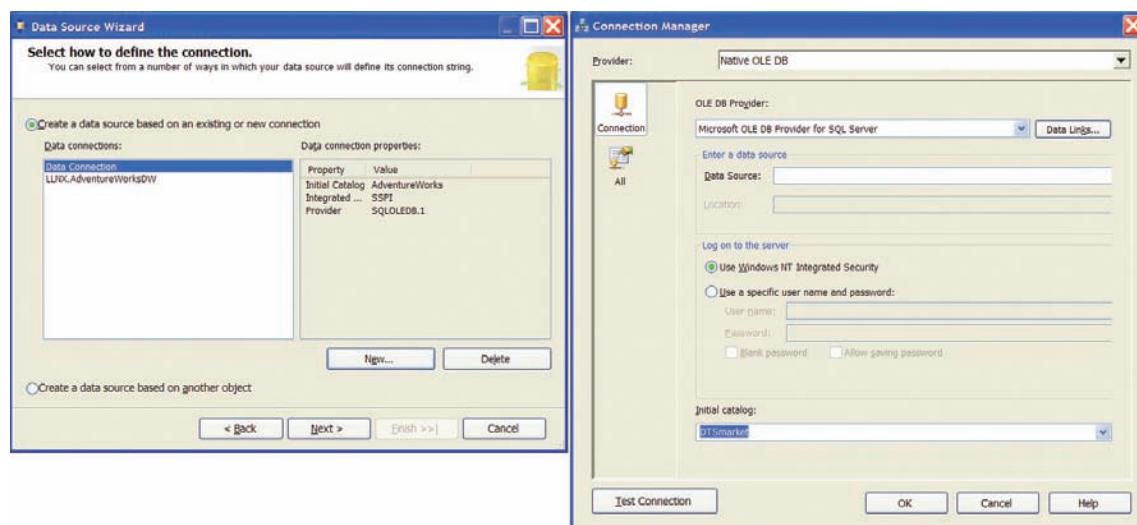
Príklad teda môžeme rozdeliť do štyroch hlavných krokov

1. definovanie dátových zdrojov
2. definovanie pohľadov na dátové zdroje
3. návrh dimenzie
4. návrh kocky

Aj keď kocka ako geometrický pojem evokuje predstavu trojdimenzionálneho priestoru, skutočné OLAP kocky majú spravidla viac dimenzií ako tri a naopak naša pravdepodobne najjednoduchšia OLAP kocka akú je vôbec možné vymyslieť má len jednu produktovú dimenziu.

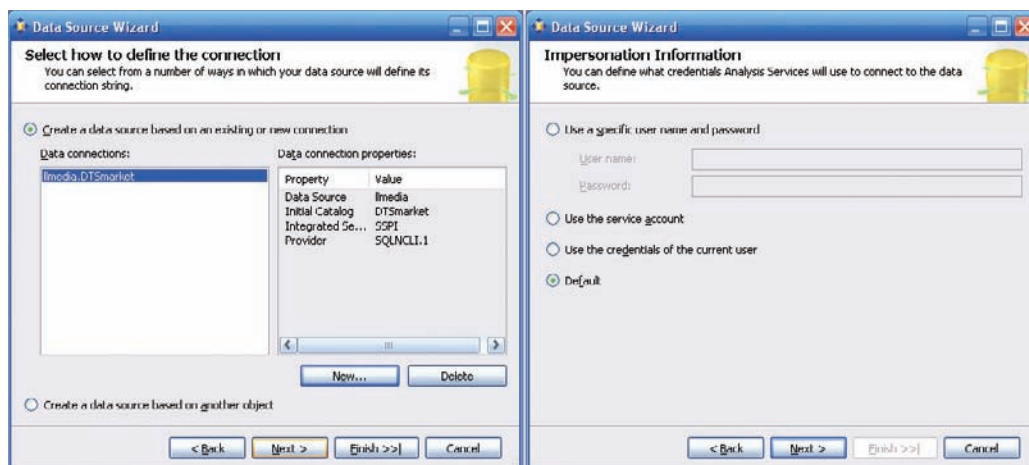
### Definovanie dátových zdrojov

Prvým krokom je definovanie zdroja údajov, ktorým je spravidla dátový sklad, prípadne jedna, alebo viacero relačných databáz. V zložke Data Sources definujeme ako dátový zdroj databázu DTSMarket. Využijeme pri tom jednoduchého sprievodcu Data Source Wizard, kde si môžeme vybrať niektoré z už nadefinovaných pripojení na databázy, alebo ako v našom prípade, vytvoríme nové pripojenie na databázu DTSMarket. Pri vytváraní pripojenia k dátovým zdrojom využijeme služby sprievodcu Data Source Wizard.



*Data Source Wizard – výber zdroja údajov*

Po definičných krokoch nového pripojenia, v našom prípade na databázu DTSMarket priradíme novo vytváranému dátovému zdroju toto pripojenie pre prístup k údajom. Data Source Wizard nám ponúka aj možnosť impersonácie, to znamená prihlásenie sa k zdroju údajov pod vlastnými používateľskými parametrami. V tomto prípade označíme v dialógu „Impersonation Information“ voľbu „Default“



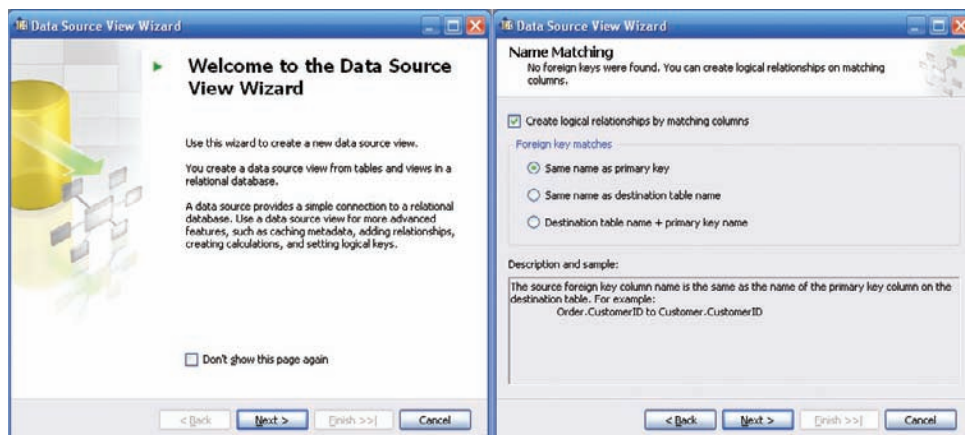
Data Source Wizard – výber zdroja údajov

Z technického hľadiska sa v procese definovania dátových zdrojov vytvoril reťazec parametrov pre pripojenie sa k príslušnému databázovému serveru

```
Provider=SQLNCLI.1;Data Source=lmedia;Integrated Security=SSPI;Initial Catalog=DTSmarket
```

### Definovanie pohľadov na dátové zdroje

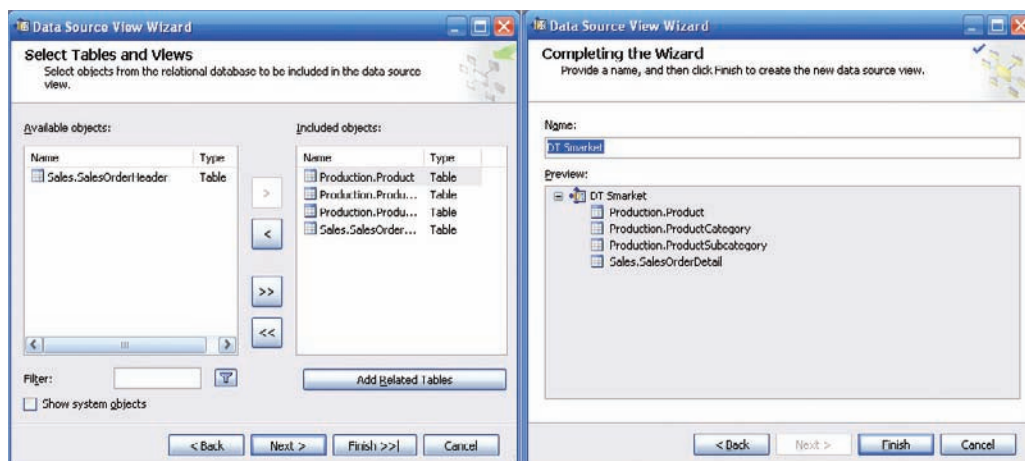
Databáza, alebo dátový sklad, ktorý poslúži ako zdroj údajov pre analýzu obsahuje veľa rôznych relačne zviazaných databázových tabuliek. Aj pre túto akciu máme k dispozícii sprievodcu – tentoraz s názvom Data Source View Wizard. V jednom z úvodných dialógov môžeme špecifikovať vytvorenie logických relačných vzťahov na základe určitých pravidiel, napríklad ak sa stĺpec v niektorej tabuľke volá rovnako ako primárny kľúč inej tabuľky a podobne.



Sprievodca pre definovanie pohľadu na dátové zdroje

V dialógu pre výber tabuliek a pohľadov vyberieme tabuľky: Sales.SalesOrderDetail, Production.Product, Production.ProductSubcategory a Production.ProductCategory – zjednodušene povedané všetky tabuľky ktoré sme v etape DTS preniesli okrem Sales.SalesOrderHeader.

Veľmi praktické je tlačidlo Add Related Tables pre pridanie relačne zviazaných tabuliek do pohľadu. Odporúčame takýto postup. Z množiny tabuliek v databáze v prípade reálneho príkladu vyberieme najskôr tabuľky faktov, v okne Included Objects ich označíme a následne tlačidlom Add Related Tables, nim budú pridelené relačne zviazané tabuľky, teda tabuľky dimenzií



Sprievodca pre definovanie pohľadu na dátové zdroje – výber databázových tabuliek pre OLAP kocku

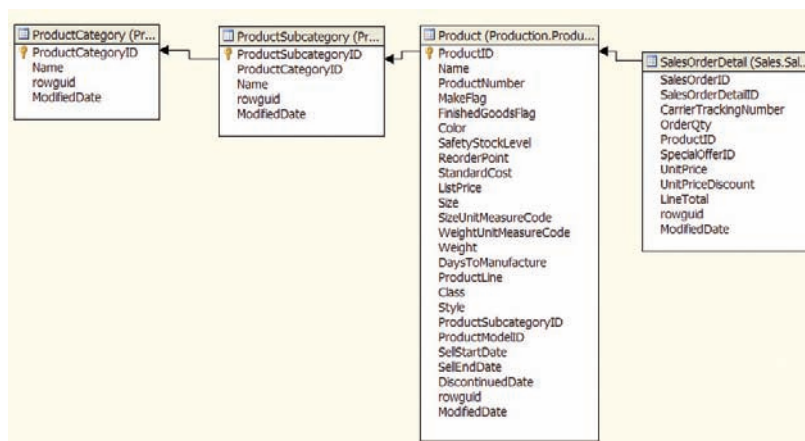
Po výbere tabuliek je potrebné nadefinovať väzby medzi nimi. Navrhujeme usporiadať v zobrazenom diagrame na pracovnej ploche tabuľky zľava doprava takto

ProductCategory --- ProductSubcategory -- Product --- SalesOrderDetail

Väzby budeme definovať zas opačne zprava doľava (podľa logiky relácií tabuliek) takto

- Medzi tabuľkami Product a SalesOrderDetail bude väzba pomocou kľúča ProductID
- Medzi tabuľkami Product a ProductSubcategory bude väzba pomocou kľúča ProductSubcategoryID
- Medzi tabuľkami ProductCategory a ProductSubcategory bude väzba pomocou kľúča ProductCategoryID

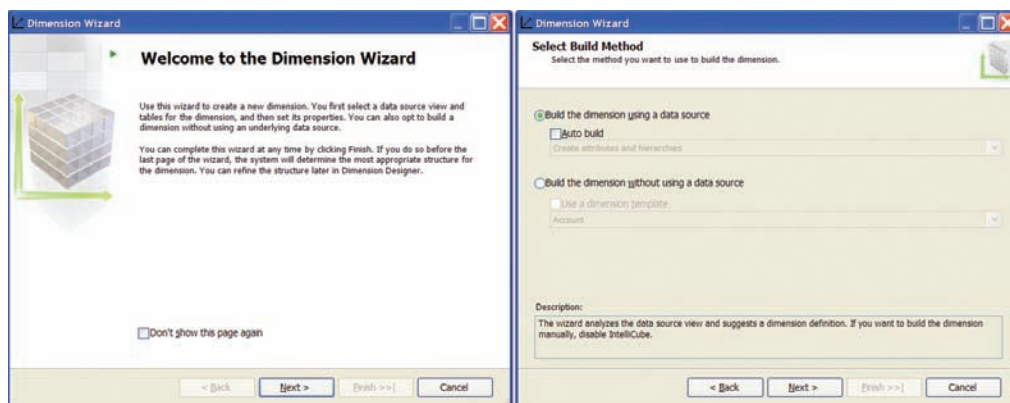
V tomto procese zároveň nadefinujeme aj primárne kľúče, nakoľko v DTS etape sme prenášali len tabuľky bez relačných väzieb. Najnázornejšie to vidíme na obrázku.



Definovanie relačných vzťahov medzi tabuľkami

### Návrh dimenzie

Asi by bolo naďalej zbytočné avizovať, že tú, alebo onú činnosť v procese vytvárania OLAP kocky nám uľahčí taký alebo onaký sprievodca. V tomto prípade to bude Dimension Wizard. Pre návrh dimenzií môžeme využiť metódu Auto Build, ktorá analyzuje dátové zdroje a navrhuje definície dimenzií, prípadne si dimenzie navrhnúť sami. V cvičnom príklade uprednostníme vlastný návrh dimenzií.



*Sprievodca pre vytvorenie dimenzií*

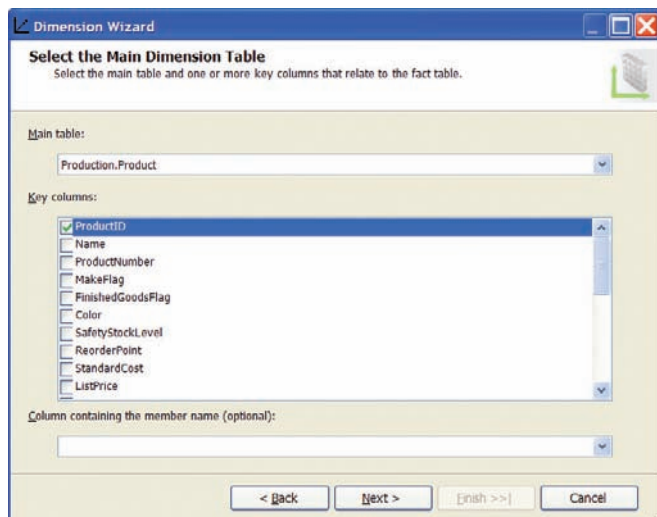
Navrhujeme produktovú dimenziu **PRODUCT** –ktorá bude mať takúto hierarchickú štruktúru

- **CATEGORY**
- • **SUBCATEGORY**
- • • **PRODUCT**

Po výbere metódy zostavenia dimenzie nasleduje výber jej typu. Dialóg sprievodcu ponúka tri typy

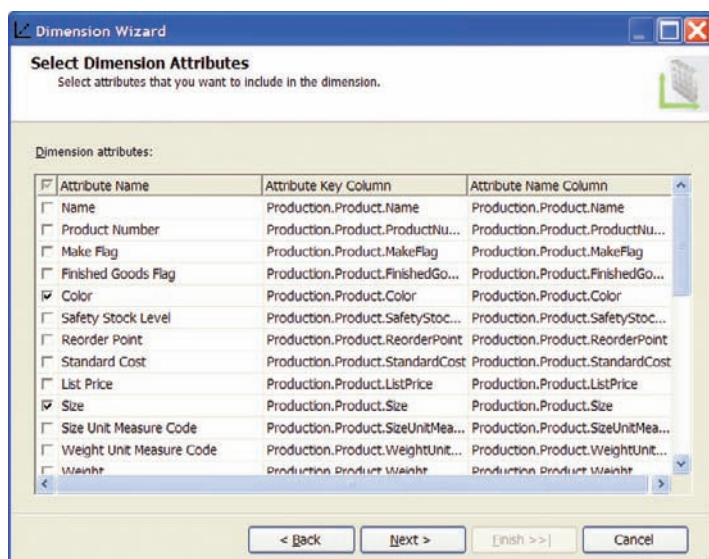
- štandardnú dimenziu
- časovú dimenziu
- časovú serverovú dimenziu

Pre produktovú dimenziu vyberieme typ štandardná dimenzia. Hlavnou tabuľkou dimenzie bude tabuľka Production.Product



*Výber hlavnej tabuľky dimenzie*

Ako kľúčový stĺpec nastavíme Product ID a stĺpec Name ako Member Column. Sprievodca v nasledujúcom dialógu podľa nami nadefinovaných cudzích kľúčov sám správne určí relačné tabuľky Subcategory a Category. Ako atribúty vyberieme stĺpce Color, Size, Product Subcategory, a Product Category

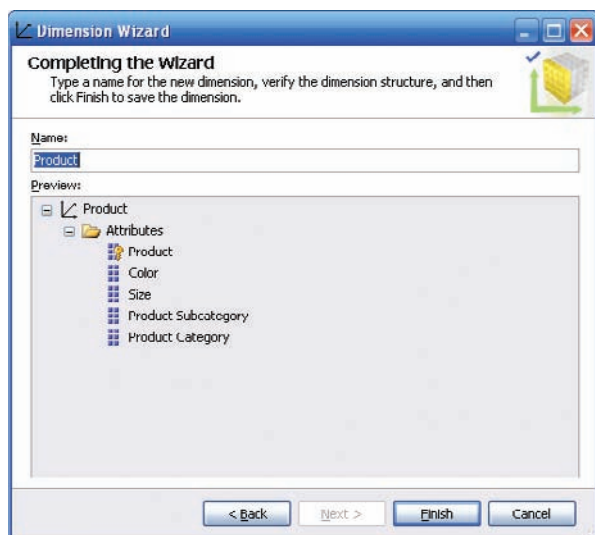


Výber relačných tabuliek a atribútov dimenzie

Pre usmernenie činnosti sprievodcu padne vhod usmernenie o aký typ dimenzie sa jedná. Preddefinované sú tieto možnosti

- Organization
- Products
- Promotion
- Quantitative
- Rates
- Regular
- Scenario
- Time

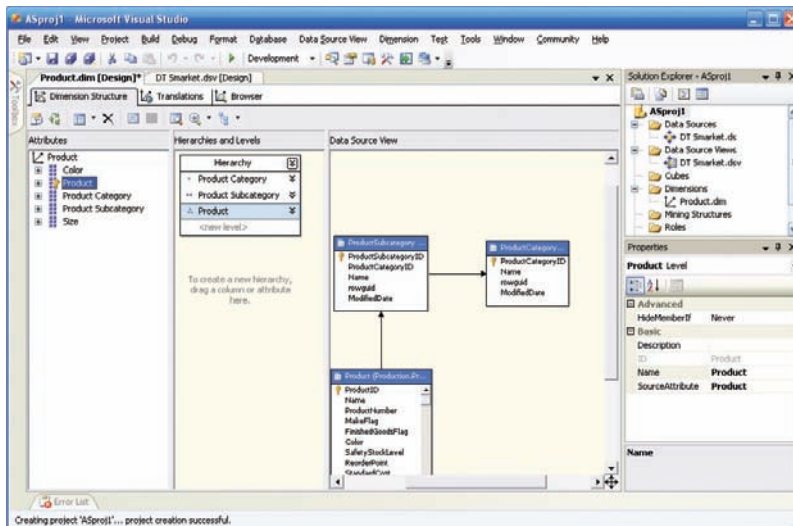
Nami vytvorená dimenzia je typu Regular bez atribútov typu parent-child.



Záverečný dialóg Sprievodcu pre vytvorenie novej dimenzie

Pomocou presúvania atribútov dimenzie navrhne jej hierarchiu

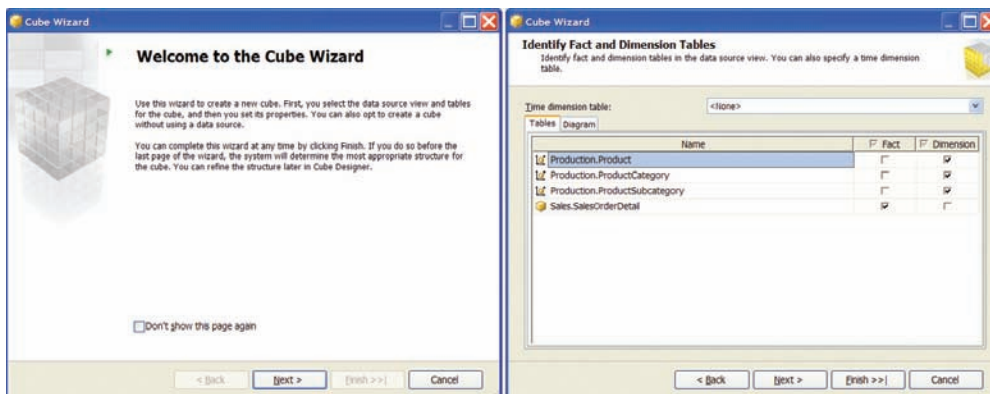
- CATEGORY
- • SUBCATEGORY
- • • PRODUCT



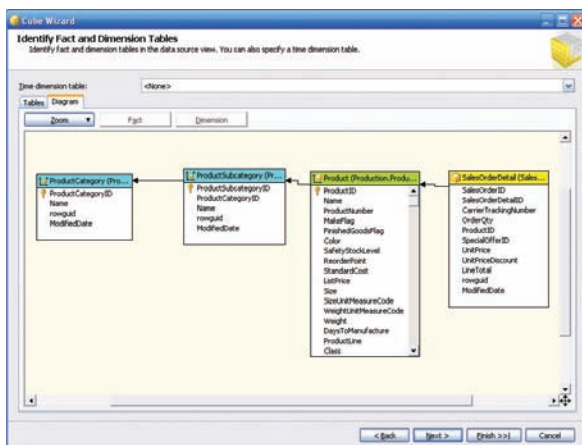
Produktová dimenzia zobrazená vo vývojovom prostredí

### Návrh kocky

Po prípravných prácach spočívajúcich v špecifikovaní údajov, ktoré budú slúžiť ako podklady pre vytvorenie tabuľky faktov a tabuliek dimenzií. Pre návrh kocky využijeme ďalšieho z rady užitočných sprievodcov – Cube Wizard. Po potvrdení zdrojov údajov prebehne automatická detekcia faktov a dimenzií, ktorá na základe analýzy relačných vzťahov medzi vybranými tabuľkami určí, ktoré tabuľky obsahujú fakty a ktoré dimenzie. Nie je to príliš náročná úloha nakoľko tabuľka faktov okrem merných jednotiek obchodovania obsahuje cudzie kľúče do tabuliek dimenzií. Sprievodcu aktivujeme pomocou položky New Cube kontextového menu zložky Cubes v okne Solution Explorera.



Výber faktov a dimenzií



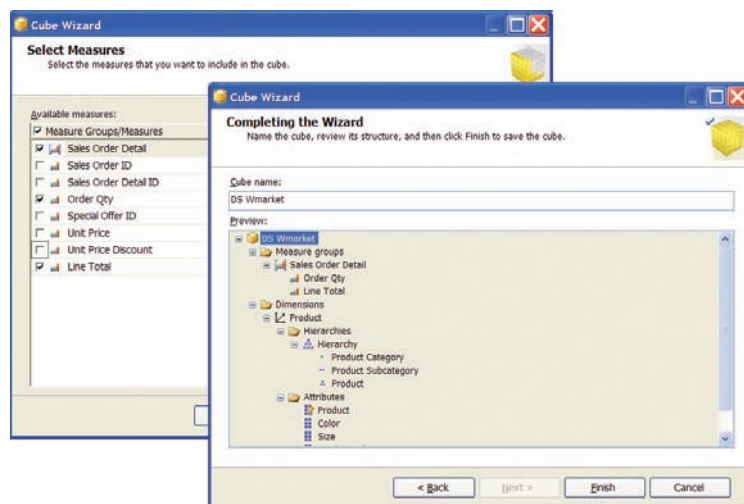
Cube Wizard – záložka diagram



Zo zoznamu dostupných dimenzií vyberieme dimenziu Product, ktorú sme vytvárali v predchádzajúcom kroku tohto príkladu.

Nasleduje výber mierok (merných jednotiek obchodovania) z tabuľky faktov. Spravidla je to počet predaných kusov, prípadne výška úhrady za tovar a služby a podobne. V tomto príklade môžeme vybrať mierky

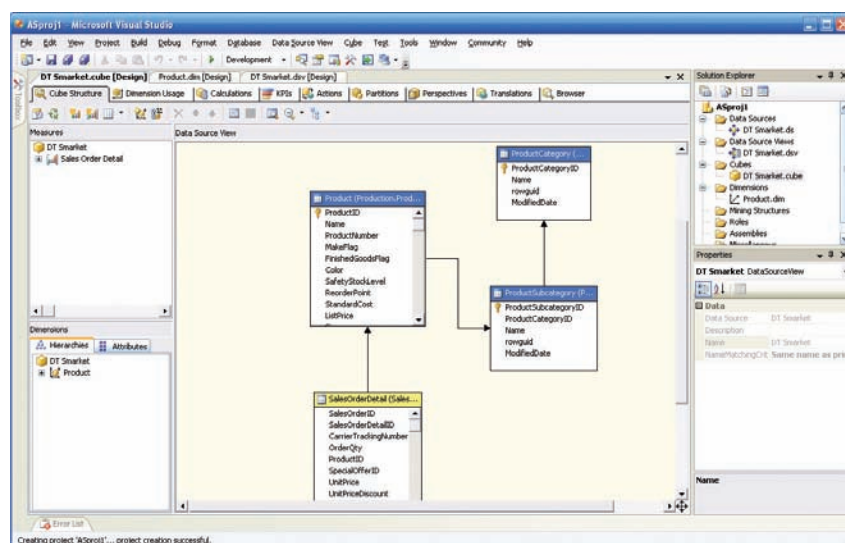
- Sales Order Detail
- Order Qty
- Line Total



Výber mierok a prehľad navrhutej kocky

Po vytvorení OLAP kocky a napočítaní agregácií sa pracovná plocha Business Intelligence Development Studia rozdelí do záložiek

- Cube Builder – záložka pre vytváranie a editovanie mierok
- Dimension Usage – pre definovanie použitia dimenzií v OLAP kockách
- Calculations – vytváranie a editovanie kalkulácií
- KPIs – Key Performance Indicators
- Actions – definovanie akcií pre navrhnutú kocku
- Partitions – prezeranie a editovanie partícií kocky
- Perspectives – budovanie perspektívnych prehľadov
- Translations – editovanie perspektívnych prehľadov
- Browser – návrh kontingenčnej tabuľky pre prezeranie údajov



Hierarchia kocky zobrazená vo vývojovom prostredí

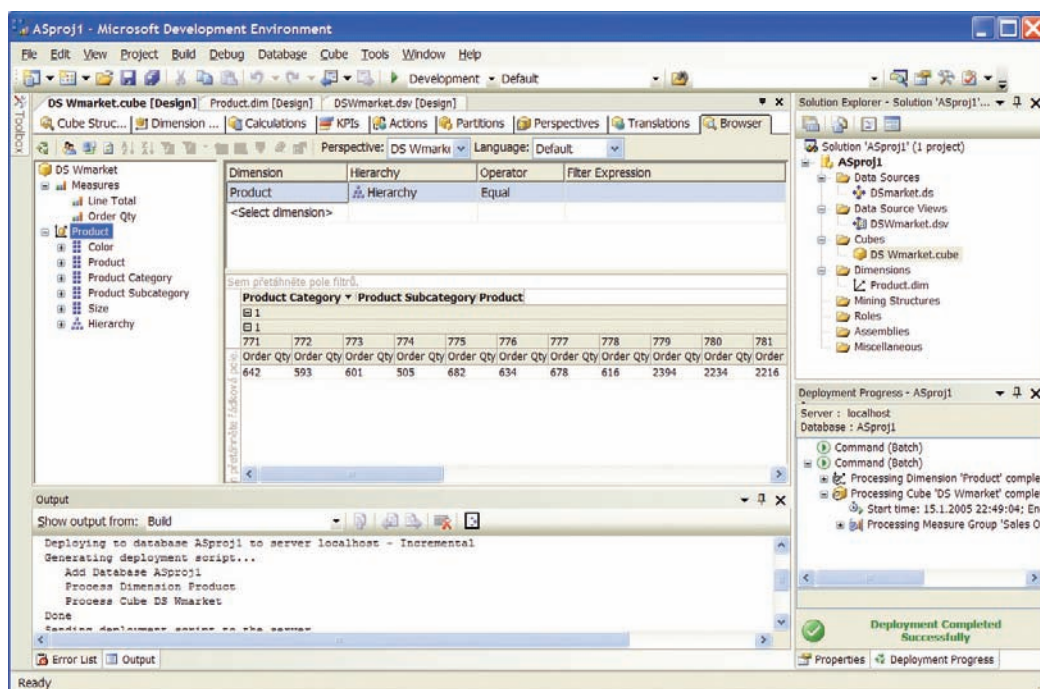
Zelenou šípkou na Toolbare zahájime zostavenie a zavedenie projektu. O priebehu tejto činnosti získame podrobný protokol

```

----- Build started: Project: ASproj1, Configuration: Development -----
Started Building Analysis Services project: Incremental ....
Build complete -- 0 errors, 0 warnings
----- Deploy started: Project: ASproj1, Configuration: Development -----
Deploying to database ASproj1 to server localhost - Incremental
Generating deployment script...
    Add Database ASproj1
    Process Dimension Product
    Process Cube DS Wmarket
Done
Sending deployment script to the server...
Done
Deploy complete -- 0 errors, 0 warnings

```

Finálnym krokom bude prehliadanie vytvorenej zostavenej OLAP kocky. V záložke Browser môžeme interaktívne vytvoriť požadovanú kontingenčnú tabuľku. Do obdĺžnika pre polia údajov umiestnime fakty a do obdĺžnikov pre polia riadkov a stĺpcov umiestnime príslušné dimenzie. Z každej dimenzie sa tak stane stáva množina polí, v ktorých možno rozbaľiť zbalit podrobnosti na jednotlivých úrovniach hierarchií.



### Prehliadanie OLAP kocky

Tento extrémne jednoduchý príklad OLAP kocky s jednou jedinou dimenziou poslužil hlavne na zoznámenie sa z návrhovými nástrojmi a postupom modelovania a návrhu.

Pre ďalšie samostatné pokusy o vytvorenie OLAP kociek z relačnej databázy výborne posluží ako zdroj údajov cvičná databáza Adventure Works. Pre vytvorenie svojich vlastných OLAP kociek môžeme napríklad tabuľku faktov *FactInternetSales* a ako podklady pre dimenzie sú určené napríklad tabuľky *DimTime*, *DimProduct*, *DimCustomer* a *DimGeography*. Aj keď ich možno pri návrhu dimenzií pre jednoduchú kocku všetky nevyužijeme, do pohľadu na dátové zdroje ich zakomponovať môžeme, prípadne môžeme skúsiť vytvoriť kocku s ešte viac dimenziami. Výsledkom nášho návrhu výberu dimenzií bude snow-flake schéma (alebo po našom schéma snehovej vločky), nakoľko dimenzia Customer pozostáva z dvoch relačne zviazaných tabuliek *DimCustomer* a *DimGeography*.

V prostredí BI Development Studio vytvoríme nový analytický projekt napríklad s názvom ASprojekt2

DimTime  
DimPromotion  
DimCurrency  
DimEmployee  
DimSalesTerritory  
FactInternetSalesReason  
DimProduct  
DimCustomer

## Vytvorenie OLAP kocky z dátového skladu

Predchádzajúci príklad mal k reálnej OLAP kocke asi tak ďaleko ako aplikácia typu „Hello World“ k reálnej aplikácii, slúžila skôr ako praktický úvod do problematiky a zoznámenie sa s vývojovým prostredím v režime BI projektov, hlavne s množinou jeho sprievodcov. OLAP kocka vytváraná v tomto prípade sa už bude realite približovať oveľa viac, bude mať viac dimenzií, pričom podobne ako u kociek v reálnej praxi jedna dimenzia bude časová. Po preštudovaní kapitoly o UDM očakávame, že tento príklad by mal mať spoločné s tým predchádzajúcim minimálne rozdelenie úkonov do logických krokov dané postupnosťou záložiek objektov vyplývajúcich s UDM

1. definovanie dátových zdrojov
2. definovanie pohľadov na dátové zdroje
3. návrh dimenzií
4. návrh kocky

Všimnite si ale pozornejšie, že poradie záložiek je iné, než poradie našich úkonov. Konkrétne sú vymenené položky „návrh dimenzií“ a „návrh kocky“

Vývojové prostredie totiž obsahuje reletívne vysoký stupeň automatizácie a pri bežných tabuľkách faktov a dimenzií sa pokúsi navrhnuť dimenzie vrátane ich hierarchie automaticky. Takže náš postup sa zjednoduší vypustením kroku pre návrh dimenzií

Okrem cvičnej transakčnej databázy AdventureWorks je spolu s SQL Serverom 2005 a jeho cvičnými príkladmi dodávaná aj cvičná databáza AdventureWorks DW. Tie dve líšiacie sa písmenká v názve (DW – Data Warehouse ) dávajú tušiť, že táto databáza je organizovaná ako typický dátový sklad. S SQL Serverom 2005 sa totiž dodávajú tri databázy AdventureWorks

- **AdventureWorks** – vzorová OLTP databáza
- **AdventureWorksDW** – vzorový dátový sklad
- **AdventureWorksAS** – vzorová analytická databáza

Najskôr je potrebné zoznámiť sa so štruktúrou dátového skladu AdventureWorksDW. Pozostáva z 23 tabuliek. Môžeme identifikovať tabuľky faktov obsahujúcich podklady pre vytvorenie mierok, teda údaje obsahujúce merné jednotky obchodovania

- FactCurrencyRate
- FactFinance
- FactInternetSales
- FactInternetSalesReason
- FactSalesQuota

a tabuliek dimenzií

- DimAccount
- DimCurrency
- DimCustomer
- DimDepartmentGroup
- DimProduct
- DimProductCategory
- DimProductSubcategory
- DimPromotion

- DimEmployee
- DimGeography
- DimOrganization
- DimSalesReason
- DimSalesTerritory
- DimTime

Určenie každej tabuľky sa dá jednoznačne identifikovať už z jej názvu. Proste vzorový dátový sklad aký je snom každého IT oddelenia.

Teraz by sme mohli obetovať dve strany publikácie na diagram dátového skladu. No nakoľko sú všetky tabuľky výstižne a logicky pomenované, zvolili sme iný postup, predpokladajúci váš aktívny prístup. Vytvorte si vo Visual Studiu 2005 jeden vedľajší projekt, v ktorom zadefinujete databázu AdventureWorksDW ako zdroj údajov. Druhý a posledný krok v tomto projekte bude zdefinovanie pohľadu na zdroje, kam zahrnieme všetky tabuľky faktov a dimenzií. V strednom okne sa zobrazí diagram, na ktorom môžeme skúmať všetky závislosti a relácie medzi tabuľkami.

Vráťme sa však k vytvoreniu analytickej databázy z dátového skladu AdventureWorksDW. Postup je podrobne popísaný v predchádzajúcom príklade.

### Definovanie dátových zdrojov

Ako dátový zdroj definujeme cvičnú databázu AdventureWorks na lokálnom, prípadne vzdialenom serveri. Využijeme sprievodcu Data Source Wizard. Pre pripojenie môžeme použiť napríklad SqlClient Data Provider zo zložky .Net Providers.

### Definovanie pohľadov na dátové zdroje

Po nadefinovaní pohľadu na dátové zdroje nastal prvý vhodný okamih na rekapituláciu. U reálnych databáz s ktorými IT oddelenia firiem denne pracujú a dôverne ich poznajú je situácia iná ako pri „cudzej“ cvičnej databáze. Diagram nadefinovaných pohľadov na dátové zdroje je vynikajúcou príležitosťou na skontrolovanie súvislostí a prípadne oboznámenie sa z „užším výberom“ zo štruktúr pôvodnej databázy.

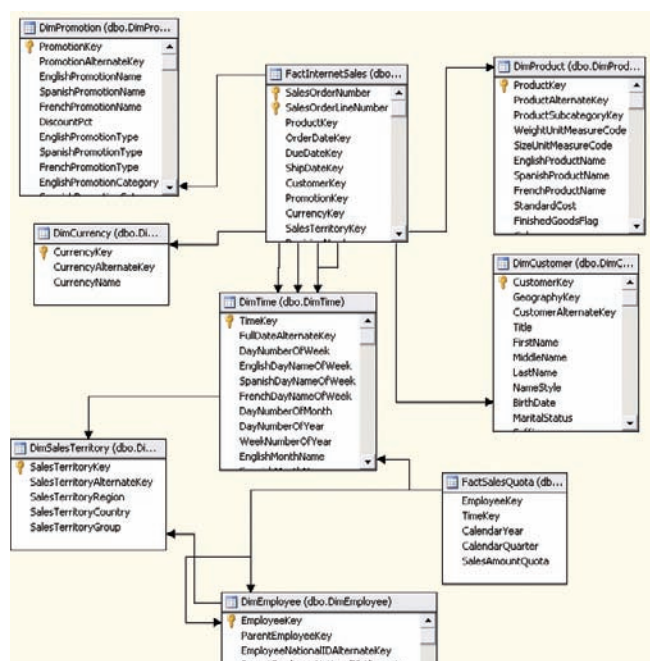


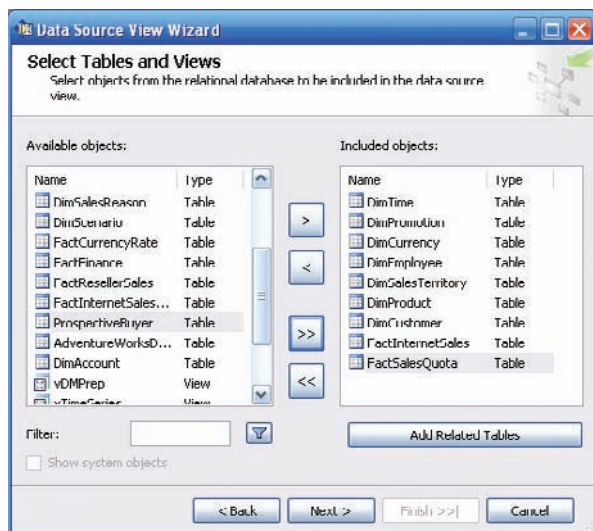
Schéma tabuľky faktov a tabuliek dimenzií

V našom príklade vyberieme tabuľky

- DimTime
- DimPromotion
- DimCurrency
- DimEmployee
- DimSalesTerritory

- DimProduct
- DimCustomer
- FactInternetSalesReason
- FactSalesQuota

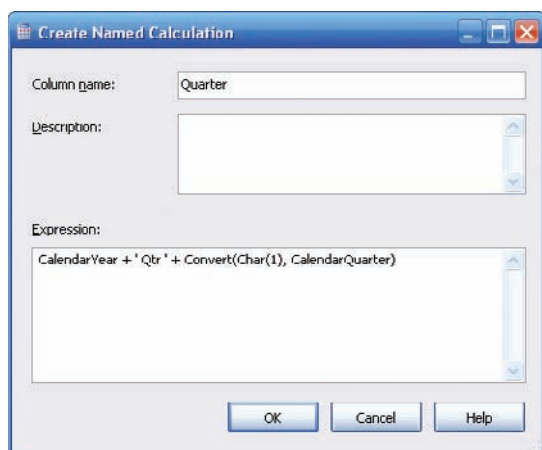
Už z ich názvov je zrejmé, že prvých sedem tabuliek obsahujú dimenzie a posledné dve tabuľky FactInternetSalesReasona FactSalesQuota obsahujú fakty



*Definovanie pohľadu na dátové zdroje – výber faktova dimenzii*

Po ukončení fázy definície pohľadov na dátové zdroje si môžeme podrobne preštudovať diagram (viď obrázok v úvode state)

Tabuľky dimenzií môžu okrem stĺpcov tabuľky obsahovať aj vypočítané atribúty. Vytvoríme si takýto atribút v tabuľke dimenzií. Kliknutím na symbol tabuľky DimTime v diagrame aktivujeme kontextové menu a v ňom položku Create Named Calculation



*Dialóg Create Named Calculation*

Ako výraz pre vypočítaný atribút napíšeme do editačného okna „Expression“

```
CalendarYear + , Qtr , + Convert(Char(1), CalendarQuarter)
```

V tomto prípade ide len o zlúčenie dvoch textových reťazcov – roku a kvartálu, medzi ktoré vložíme skratku Qtr. Pýtate sa prečo táto redundancia? Nuž keď si budú analytici vytvárať z tejto dimenzie rôzne kocky, aby mali



V procese detekcie hierarchií dimenzií budú v prvom kroku procesu vypísané dimenzie.

- DimTime
- DimPromotion
- DimCurrency
- DimEmployee
- DimSalesTerritory
- DimProduct
- DimCustomer

V tomto kroku môžeme niektoré dimenzie zatiaľ vynechať, napríklad pre tento príklad, kedy budeme analyzovať odbyť a nie výrobu nebudeme potrebovať dimenziu DimEmployee. V definícii pohľadu na dátové zdroje však tabuľka obsahujúca údaje tejto dimenzie zostáva a dimenziu môžeme kedykoľvek dodefinovať, napríklad v prípade ak budeme analyzovať výrobu. Nezabudnime špecifikovať časovú dimenziu.

Veľmi názorne zobrazuje situáciu prehľadný diagram, kde sú tabuľky faktov a dimenzií farebne odlišené a sú v ňom znázornené aj všetky relačné väzby

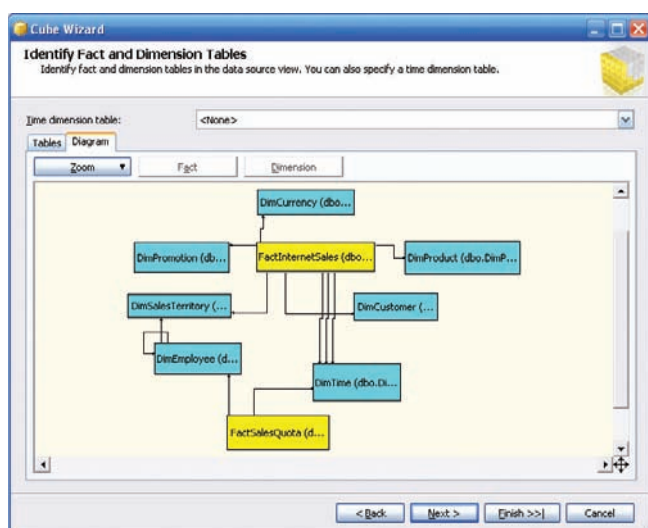
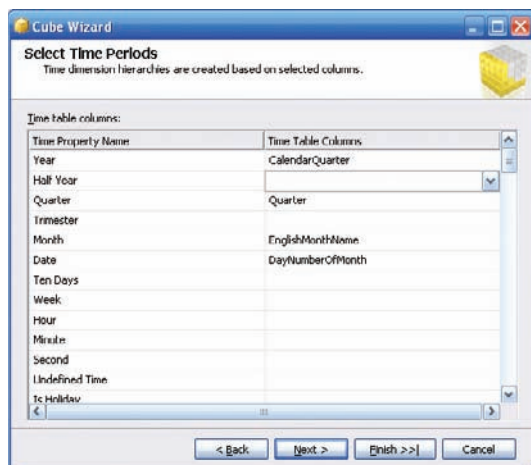


Diagram výsledku automatickej identifikácie faktov a dimenzií

Prácu s časovou dimenziou sme ešte neskončili. Po označení tabuľky Dim Time je v ďalšom dialógu „Select Time Periods“ potrebné špecifikovať, ktorý atribút tabuľky obsahuje údaje o akom časovom období. Aj napriek tomu, že názvy (Year, Quarter, Month, Day...) v angličtine by mohli byť pre sprievodcu pri určovaní časových období určitým vodítkom, pri explicitnom určení časových období si sprievodca vytvorením kocky poradí aj s názvami atribútov v ľubovoľnom jazyku, prípadne názvami inak organizovanými, napríklad aj s nič nehovoriacimi názvami Column1, Column2....

V dialógu Select Time Periods označíme časové periódy napríklad takto:

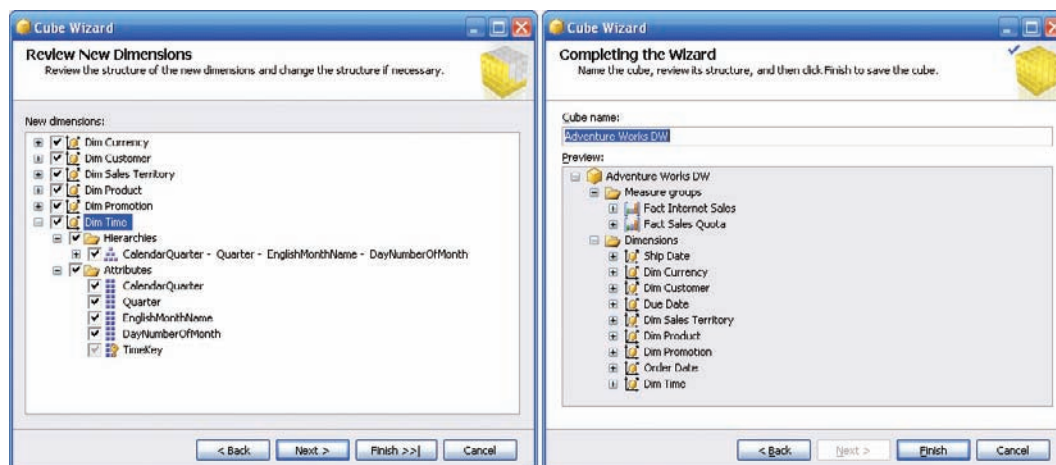
Time Property Name	Time Table Columns
Year	CalendarYear
Quarter	Quarter
Month	EnglishMonthName
Date	DayNumberOfMonth



Dialóg pre definovanie períód časovej dimenzie

V tabuľke mierok môžeme ponechať všetky atribúty navrhnuté sprievodcom, okrem atribútu „RevisionNumber“, u ktorého je už podľa názvu jasné, že neobsahuje údaje o obchodovaní.

Proces pokračuje automatickým určením hierarchickej štruktúry dimenzií. Výsledok je zobavený v dialógu „Review New Dimensions“.



Výsledok identifikácie hierarchie dimenzií

U každej dimenzie si môžeme v stromovej štruktúre prezrieť jej atribúty a hierarchiu. V druhej fáze identifikácie dimenzie si určite všimnete, že pôvodne jedna časová dimenzia bola rozdelená na tri nezávislé dimenzie

- Dim Product
- Dim Customer
- Due Date
- Order date
- Ship Date

Dôvod je jednoduchý. Všimnite si na schéme, že väzba medzi tabuľkou faktov a tabuľkou časovej dimenzie je realizovaná pomocou troch cudzích kľúčov DueDateKey, OrderDateKey a ShipDateKey. Výhodou takéhoto prístupu je do značnej miery obmedzenie redundancie, nakoľko pre tri prehľadné časové dimenzie postačí jedna a tá istá databázová tabuľka.



## Zostavenie analytickej databázy

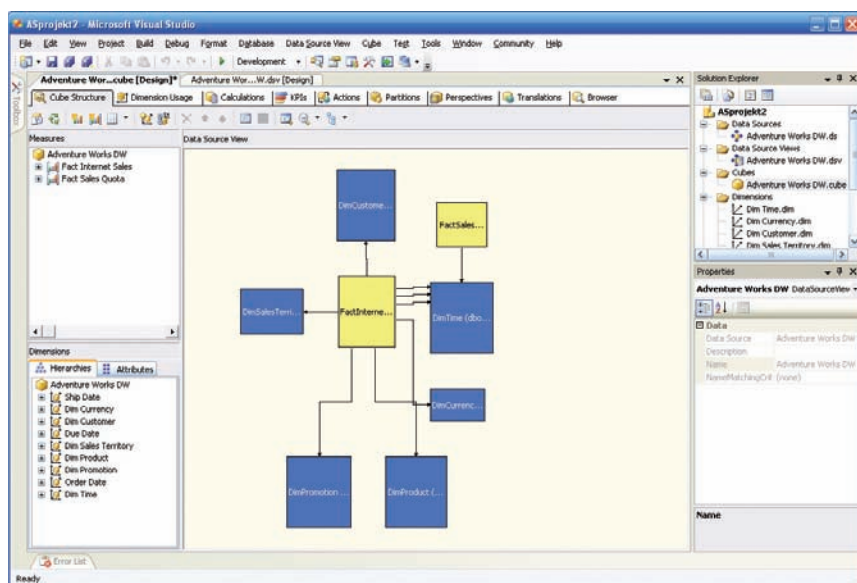
Po ukončení návrhu vykonáme deploy aplikácie pomocou zeleného tlačidla v tvare šípky na toolbare

## Práca s OLAP kockou v prostredí BI Dev Studio

Zatiaľ čo v predchádzajúcom dalo by sa povedať „zoznamovacom“ príklade sme kladli dôraz na postup pre modelovanie OLAP štruktúry v tomto príklade to bude naopak. Postup vytvorenia OLAP kocky z efektívne usporiadaného dátového skladu sme zhrnuli do niekoľkých heslovitých bodov a o to väčšiu pozornosť budeme venovať možnostiam práce s OLAP kockou v prostredí BI Dev Studia. Postupne si prezrieme jednotlivé záložky

### Cube Builder

V záložke je možné prezerat' a upravovat' model OLAP kocky. V prehľadnom diagrame sú znázornené relačné vzťahy medzi tabuľkami faktov a dimenzií.

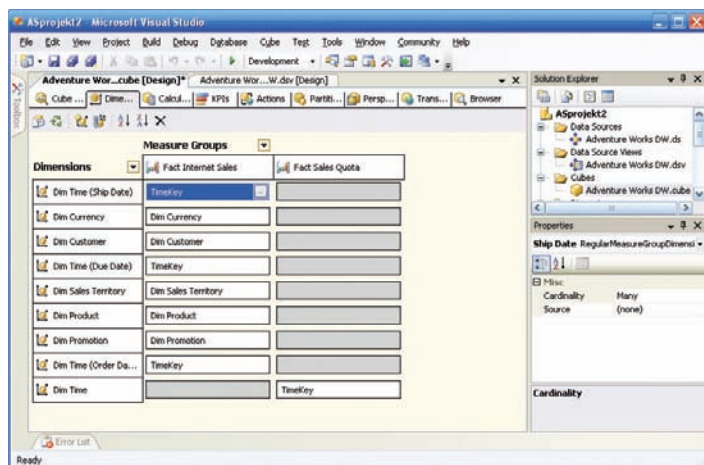


Model OLAP kocky vo vývojovom prostredí

Jednou z užitočných záležitostí, ktoré je možné nastavovat' v tomto okne je formát zobrazenia jednotlivých mierok.

### Dimension Usage

Záložky slúžia pre definovanie použitia dimenzií v OLAP kockách, prípadne editovanie relačných vzťahov medzi tabuľkami faktov a dimenzií. V samostatných dialógoch pre jednotlivé dimenzie, ktoré sa aktivujú tlačidlom (...) je možné okrem typov relačných vzťahov (Regular, Fact, Referenced, Many-to-many) meniť aj granularitu jednotlivých atribútov.



Záložka Dimension Usage

Funkciu ostatných záložiek odporúčame podrobne preskúmať, preštudovať dokumentáciu a vyskúšať si možnosti, ktoré ponúkajú

### Browser

Po úspešnom zostavení OLAP kocky a jej zavedení na analytický server nastala prvá príležitosť na prehliadnutie a skontrolovanie výsledkov analýzy. Nie je náhoda, že rozmiestnenie pracovného okna je veľmi podobné ako pri návrhu kontingenčnej tabuľky pre prezeranie údajov v programe Excel. Vývojové prostredie totiž využíva komponentu Office Web Components. Okrem plochy v strede tabuľky pre údaje, kam umiestnime mierky sú v návrhovom liste kontingenčnej tabuľky ešte polia

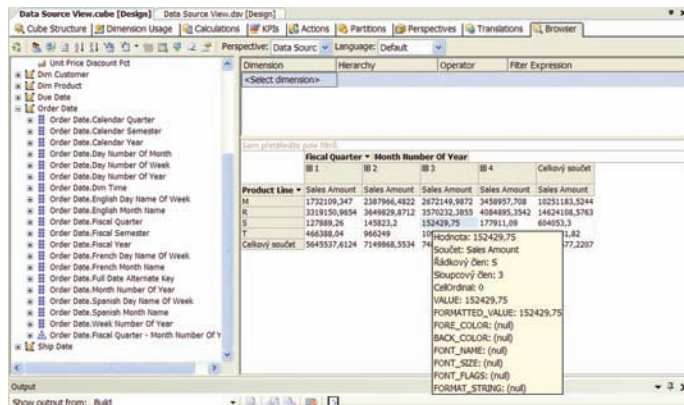
Riadkové polia

Stĺpcové polia

Polia filtrov

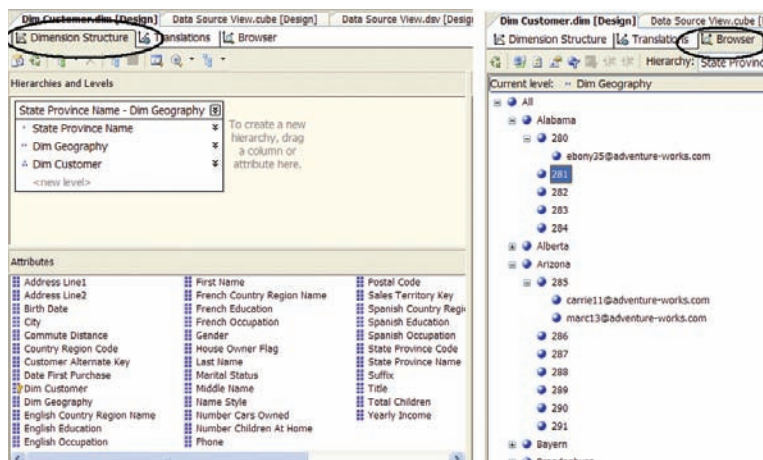
kam spravidla umiestňujeme dimenzie.

Ako prvý krok pre zamýšľanú zostavu preniesime symbol mierky Sales Amount z ľavého okna Data Source View do vnútra kontingenčnej tabuľky. Do riadkových polí presunieme symbol Product line (atribút produktovej dimenzie). Do stĺpcového poľa preniesime celú hierarchiu dimenzie OrderDate (symbol OrderData.Fiscal Quarter).



Prehliadanie údajov OLAP kocky

Ak nie sme spokojní s formátom zobrazovania, tento môžeme zmeniť v záložke Cube Structure. Ak v ktorejkoľvek záložke a v okne Solution Explorer označíme ľubovoľnú dimenziu, môžeme si prehliadať jej štruktúru aj hierarchiu.

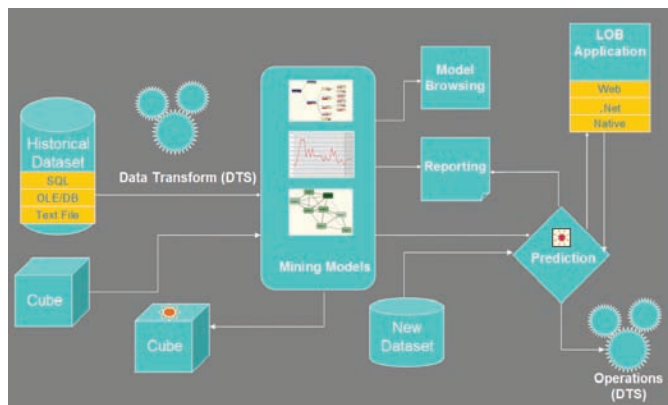


Prehliadanie dimenzie

Tu môžeme meniť úrovne hierarchie, prípadne atribúty na jednotlivých úrovniach. Stačí interaktívne presúvať atribúty medzi oknami Hierarchies and Levels a Attributes.

## Kapitola 5: Dolovanie údajov – data mining

Data mining je momentálne jednoznačne najrýchlejšie rastúci segment Business Intelligence a v súčasnosti podobne ako OLAP analýzy majú túto technológiu implementované všetky významné komerčne dodávané databázové servery. S trochou nadsázky by sa dalo povedať, že sa jedná o odvetvie niekde na rozhraní vedy a mágie. Ako vyplýva z názvu, ide v princípe o ťažbu údajov, presnejšie o proces analýzy dát z rôznych perspektív ich premenu na užitočné informácie. Z matematického a štatistického hľadiska ide o hľadanie korelácií, teda vzájomných vzťahov alebo vzorov v údajoch. Na základe nazbieraných údajov sa pokúšame zisťovať (odkrývať) rôzne závislosti, ktoré tieto údaje v sebe skrývajú. Data mining umožňuje vyhľadávať vzory informácií v údajoch. Je založený na heuristických algoritmoch, umelej inteligencii, neurónových sieťach iných pokročilých databázových softvérových technológiách a metódach umelej inteligencie. Technológie pre dolovanie údajov pomáhajú jednak sledovať a analyzovať trendy a taktiež predvídať udalosti. V ekonomickej oblasti sa data mining využíva napríklad pri analýze a predikcii úverového rizika, predikcie rizika pri vydávaní kreditných kariet. Čoraz častejšie sa však výhody tejto technológie využívajú aj v iných oblastiach, napríklad pri analýze laboratórnych vzoriek, v medicíne a podobne. V niektorých prípadoch býva technológia dolovania údajov označovaná aj skratkou KDD (Knowledge discovery in databases)



Integrácia data miningu do BI procesov

### Procesná schéma data miningu

Proces data miningu je možné rozčleniť do troch etáp.

- výber algoritmu a modelu,
- učiaci fáza aplikovaná na doteraz existujúcich prípadoch,
- analýza a predikcia nových prípadov.

*Dm1.bmp Obr. X Procesná schéma data miningu*

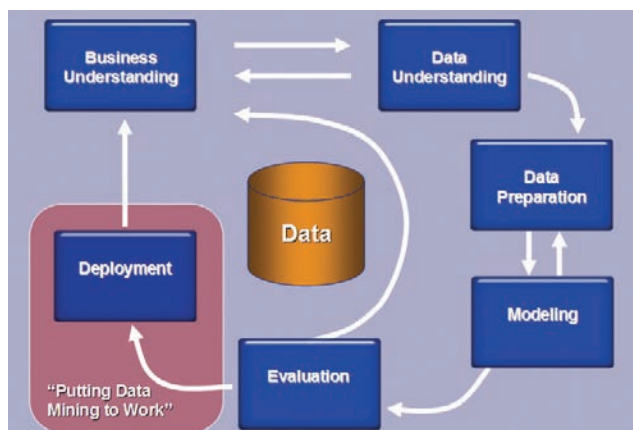
Alebo inými slovami, na základe prieskumu údajov sa v nich vyhľadávajú určité vzory a na základe týchto vzorov sa vypracujú modely pre predikciu budúcich prípadov. Pochopiteľne vzory sa podarí odhaliť len vtedy, ak nejaké existujú. Preto napríklad nemá význam aplikovať data mining nad rôznymi cvičnými databázami obsahujúcimi náhodne vygenerované údaje.

### Učiaci fáza

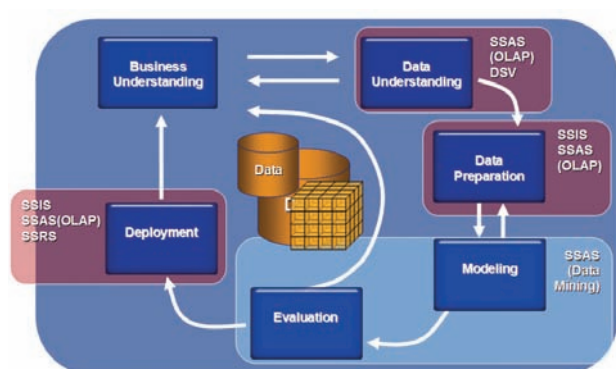
V tejto fáze sa vybraný data miningový model učí a spresňuje svoje parametre na množine údajov získaných z doteraz existujúcich prípadov. Ako podklad pre učiacu fázu slúžia doteraz zozbierané (namerané) a vyhodnotené údaje. Výberu a príprave údajov, ktoré budú slúžiť ako podklad pre učiacu fázu musíme venovať veľkú pozornosť, aby boli dostatočne reprezentatívne a očistené od prípadných chýb. Najvýhodnejšie je pripraviť a predspracovať tieto údaje v procese ETL. Vidíme, že učiaci fáza je veľmi náročná na výber správneho modelu a taktiež na výber a predspracovanie údajov.

### Analýza a predikcia nových prípadov

V etape analýzy a predikcie aplikujeme už naučený data miningový model na množinu vstupných údajov, z ktorých potrebujeme získať súvislosti. Na rozdiel od učiacej fázy je táto fáza skôr náročná na výpočtovú kapacitu použitého hardvéru a softvéru.



Procesná schéma data miningu údajov z relačnej databázy



Procesná schéma data miningu údajov z OLAP kocky

### Algoritmy pre data mining

Data mining je proces analýzy dát z rôznych perspektív a ich premena na užitočné informácie. Z matematického a štatistického hľadiska ide o hľadanie korelácií, teda vzájomných vzťahov alebo vzorov v údajoch. Data miningová časť analytických služieb bola obohatená o nové algoritmy. Zatiaľ, čo SQL Server 2000 obsahoval len algoritmus hľadania zhukova nevyvážené rozpadové stromy, SQL Server 2005 má implementovaných sedem algoritmov

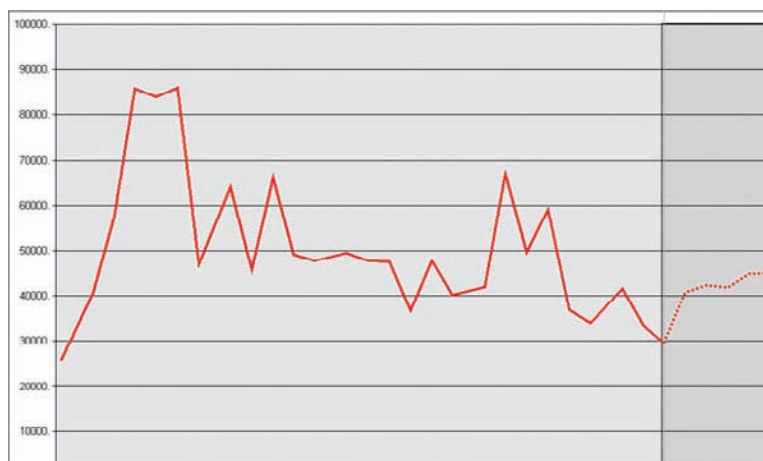
- Asociačné pravidlá
- **Nevyvážené rozpadové stromy**
- **Zhlukovanie (clustering)**
- Naïve Bayes
- Neuronové siete
- Sekvenčné zhukovanie
- Časové série

príčom aj algoritmy známe z predchádzajúcej verzie (vytlačené hrubým fontom) sú značne vylepšené. Jednoznačné odporúčanie pre výber najvodnejšieho algoritmu pre tú ktorú úlohu neexistuje, vždy to závisí od každého konkrétneho prípadu. Pre uľahčenie orientácie najskôr popíšeme jednotlivé algoritmy a potom sa pokúsime priradiť vhodné algoritmy k jednotlivým typom úloh. Pri popise algoritmov uvádzame aj zoznam parametrov, pomocou ktorých špecifikujeme úlohu, ktorú chceme pomocou algoritmu riešiť

### Asociačné pravidlá

Analýza z hľadiska asociačných pravidiel je zameraná na odhaľovanie rôznych súvislostí priradovania. Najčastejšie sa pravdepodobne používa pri hľadaní odpovede na otázku, aké tovary si zákazníci kupujú najčastejšie spolu. Poznanie týchto súvislostí umožní efektívnejšie rozmiestnenie tovarov v supermarketoch, prípadne zostavovanie akciových ponúk. Z hľadiska matematickej štatistiky sa jedná o skúmanie korelácie, či už pozitívnej, alebo

negatívnej. Pozitívna korelácia udáva, že vysoká úroveň jednej premennej bude sprevádzaná vysokou úrovňou korelačnej premennej. Naopak negatívna korelácia udáva, že vysoká úroveň jednej premennej bude sprevádzaná nízkou úrovňou korelačnej premennej. Poznanie pozitívnej korelácie je dôležité napríklad pri rozmiestňovaní tovaru alebo pri marketingových rozhodnutiach, ktoré tovary je možné predávať spolu, prípadne čo je potrebné ktorému zákazníkovi ponúknuť. Napríklad v supermarketoch sa zistilo, že sa pomerne často nakupujú spolu detské plienky a pivo. „Spotrebiteľ“ plienok teda batola je však v tomto prípade mimo podozrenia. Túto koreláciu majú na svedomí nákupcovia. Veľké balenia plienok obvykle nakupujú oteckovia, pretože mamička sa venuje potomkovi a po identifikovaní tohto faktu už nákup piva spolu s plienkami nie je žiadnou záhadou. Význam však má aj negatívna korelácia, napríklad pri výmene skladby firemnej produkcie. Môžeme to demonštrovať napríklad záujmom výrobcov klasických fotoaparátov o fotoaparáty digitálne.



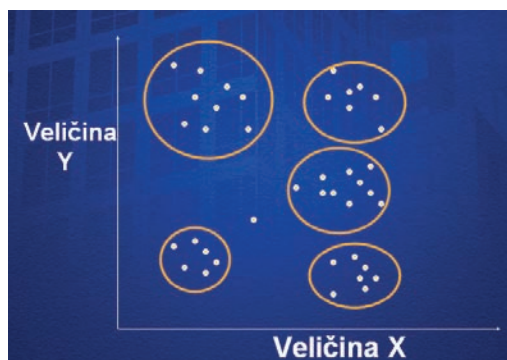
Asociačné pravidlá.

#### Parametre

MINIMUM\_SUPPORT, MAXIMUM\_SUPPORT, MAXIMUM\_ITEMSET\_COUNT, MAXIMUM\_ITEMSET\_SIZE, MINIMUM\_ITEMSET\_SIZE, MINIMUM\_PROBABILITY

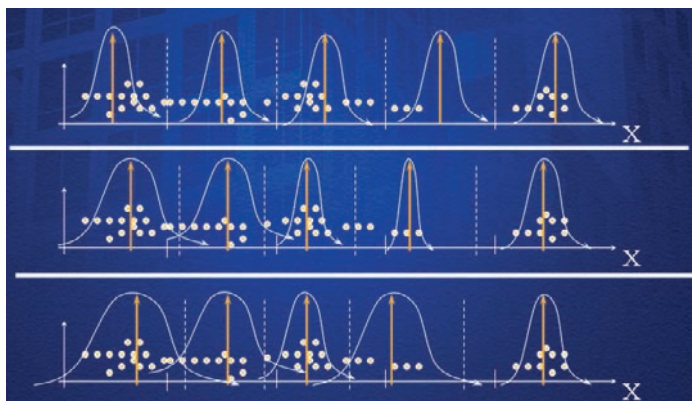
#### Viacrozmerné zhlukové diagramy

Tento algoritmus sa používa pre odhaľovanie zhlukov dát v multidimenzionálnych priestoroch. Pomocou neho môžeme rozdeliť množiny prípadov na čo najohraničenejšie skupiny, takzvané „ostrovy podobnosti“. Vhodný je napríklad pre identifikáciu zákazníckych segmentov, ktoré sú založené na spoločných charakteristikách, napríklad demografických, sociálnych, profesijných a podobne.

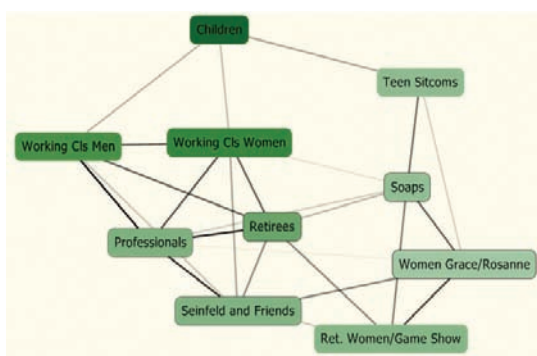


Odhaľovanie zhlukov

Zhluky sú odhaľované na základe aplikovania a analýzy kriviek pravdepodobnosti, pričom sa skúma, či jednotlivé údaje, alebo skupiny údajov nespĺňajú podmienku napríklad Gaussovho normálneho rozdelenia a podobne



Princíp vyhľadávania segmentov.



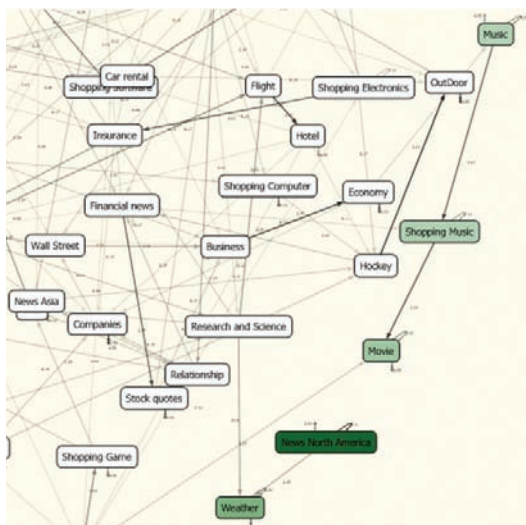
Zhlukovanie.

**Parametre**

CLUSTER\_COUNT, MAXIMUM\_INPUT\_ATTRIBUTES, CLUSTER\_METHOD, MAXIMUM\_STATES, MINIMUM\_CLUSTER\_CASES, MODELLING\_CARDINALITY, STOPPING\_TOLERANCE

**Sekvenčné zhlukovanie**

Špecifickým prípadom zhlukovania je sekvenčné zhlukovanie. Z matematického hľadiska sa jedná o aplikáciu Markových procesov a modelov. Na rozdiel od asociačných pravidiel u sekvenčného zhlukovania záleží na poradí prípadov.



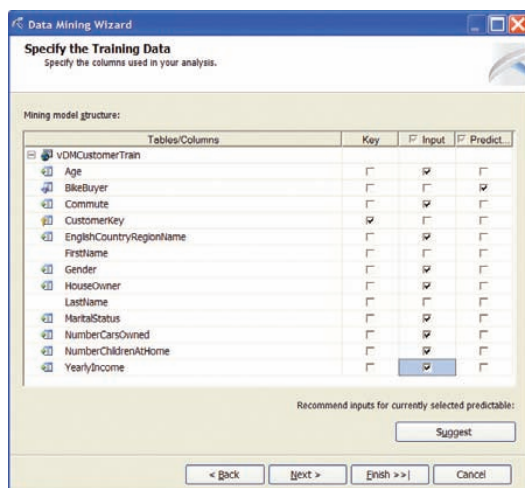
Sekvenčné zhlukovanie.

**Parametre**

CLUSTER\_COUNT, MINIMUM\_CLUSTER\_CASES, MAXIMUM\_STATES,  
MAXIMUM\_SEQUENCE\_STATES

**Nevyvážené rozhodovacie stromy**

Tento typ diagramov odhaľuje závislosti a vyhľadáva špecifické vlastnosti, ktoré potom poslúžia pre zostavenie predikčného modelu rozhodovania na jednotlivých úrovniach hierarchickej štruktúry..



*Nevyvážené rozpadové stromy.*

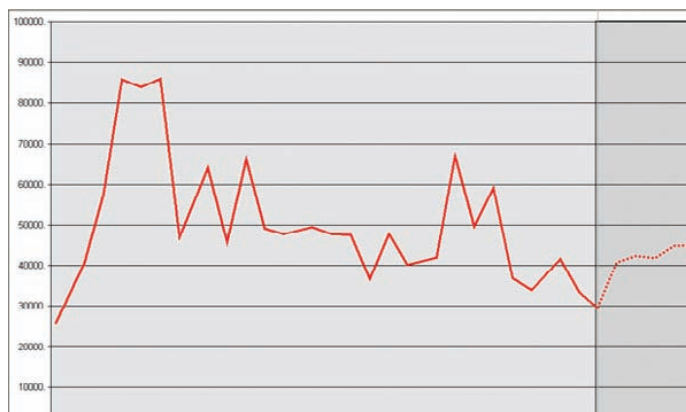
**Parametre**

COMPLEXITY\_PENALTY, MAXIMUM\_INPUT\_ATTRIBUTES, MAXIMUM\_OUTPUT\_ATTRIBUTES,  
MINIMUM\_LEAF\_CASES, FORCE\_REGRESSORS, SCORE\_METHOD, SPLIT\_METHOD

Algoritmus podporuje predikciu diskretných aj spojitých atribútov.

**Časové rady**

Algoritmus analýzy časových radov sa spravidla vzťahuje k nejakej premennej, napríklad obratu, zisku, nákladov a podobne. Na základe analýzy údajov z minulosti je možné definovať určité pravidlá, pomocou ktorých potom predpovedáme budúci trend danej premennej. V princípe ide o regresiu časových úsekov, takže predpovede trendov v sebe zahrňujú aj krátkodobé cyklické fluktuácie, napríklad ak predikujeme trend vývoja nezamestnanosti pre budúci rok, na základe údajov z minulých rokov, určite sa v dobrej predpovedi objavia aj logické fluktuácie, napríklad nárast nezamestnanosti po ukončení školskej dochádzky, jej pokles pri sezónnych prácach podobne



*Časové rady.*

**Parametre**

AUTO\_DETECT\_PERIODICITY, HISTORIC\_MODEL\_COUNT, HISTORIC\_MODE\_GAP, COMPLEXITY\_PENALTY, MINIMUM\_LEAF\_CASES, MISSING\_VALUE\_SUBSTITUTION, PERIODICITY\_HINT

**Neurónové siete** Spracovanie pomocou neurónových sietí nevychádza zo žiadneho štatistického rozdelenia ale pracuje na princípe rozpoznávania vzorov a minimalizácie chýb. Tento proces si môžeme predstaviť ako prijímanie informácií a poučenie sa z každej skúsenosti. Neurónová sieť tvoria uzly usporiadané do vrstiev. Predtým, než sa začne vlastný proces, údaje sa rozdelia do trénujacej a testovacej množiny. Počas každej iterácie sú vstupy spracovávané systémom a sú porovnávané so skutočnou hodnotou. Zmeria sa chyba a odovzdá sa k spracovaniu systému, aby upravil pôvodné váhy. Proces sa končí spravidla v okamihu dosiahnutia dopredu určenej minimálnej chyby. Obrázok ilustruje jednoduchú neurónovú sieť z jednou skrytou vrstvou.

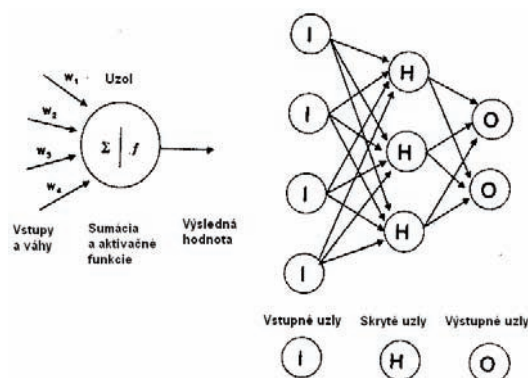


Schéma neurónovej siete

**Parametre**

MAXIMUM\_INPUT\_ATTRIBUTES, MAXIMUM\_OUTPUT\_ATTRIBUTES, MAXIMUM\_STATES, HIDDEN\_NODE\_RATIO, HOLDOUT\_PERCENTAGE,

Algoritmus siete podporuje predikciu diskretných aj spojitéch atribútov, no najvýhodnejšie je použiť len diskretné alebo diskretizované atribúty.

**Naïve Bayes**

Tento veľmi rýchly a pritom pomerne presný algoritmus je založený na Bayesovej vete, pomocou ktorej je možné pracovať priamo s pravdepodobnosťami. Pre svoju rýchlosť sa využíva hlavne na rýchlu predbežnú analýzu, pomocou ktorej vylúčime nedôležité atribúty. Potom sa spravidla aplikuje časovo náročnejší algoritmus (rozpadové stromy, neurónové siete...)

Pravdepodobnosť nastania náhodnej udalosti A za podmienky, že nastala náhodná udalosť B sa rovná podielu pravdepodobnosti prieniku náhodných udalostí A a B a pravdepodobnosti B.

Teda platí

$$P(A \cap B) = P(A|B) * P(B)$$

$$P(B \cap A) = P(B|A) * P(A)$$

$$P(A \cap B) = P(B \cap A) \Rightarrow P(A|B) * P(B) = P(B|A) * P(A)$$

Bayesovu vetu potom môžeme zapísať v tvare

$$P(A|B) * P(B) = P(B|A) * P(A)$$

Tento algoritmus sa používa pre zložitejšie analýzy, napríklad pre výpočet paralelnej korelácie a podobne



Discrimination scores for Professional/Technical and Service Workers			
Attributes	Values	Favors Professional/Techn...	Favors Service Workers
Education Years	15 - 20		
Education Years	12 - 13		
Education Years	7 - 12		
nielson hits(YOUNG AND THE RES...	Missing		
nielson hits(YOUNG AND THE RES...	Existing		
nielson hits(AS THE WORLD TURN...	Existing		
nielson hits(AS THE WORLD TURN...	Missing		

Naive Bayes

### Parametre

MAXIMUM\_INPUT\_ATTRIBUTES, MAXIMUM\_OUTPUT\_ATTRIBUTES,  
MINIMUM\_NODE\_SCORE, MAXIMUM\_STATES,

Algoritmus podporuje predikciu len diskretných alebo diskretizovaných atribútov.

Viac informácií data miningových algoritmov môžete získať na adresách

Decision trees (classification/regression):

- <ftp://ftp.research.microsoft.com/users/surajitc/icde99.pdf>
- [http://www.research.microsoft.com/research/pubs/view.aspx?tr\\_id=81](http://www.research.microsoft.com/research/pubs/view.aspx?tr_id=81)
- <http://research.microsoft.com/~dmax/publications/dmart-final.pdf>

Clustering

- EM: [http://www.research.microsoft.com/scripts/pubs/view.asp?TR\\_ID=MSR-TR-98-35](http://www.research.microsoft.com/scripts/pubs/view.asp?TR_ID=MSR-TR-98-35)

Sequence clustering

- <ftp://ftp.research.microsoft.com/pub/tr/tr-2000-18.pdf>

Time series:

- <http://research.microsoft.com/~dmax/publications/dmart-final.pdf>

### Typické okruhy úloh a výber algoritmov pre ich riešenie

Výber algoritmov pre riešenie typických okruhov úloh nie je jednoduché a už zďaleka nie jednoznačné.

Vymenovanie a stručný popis algoritmov nám nie vždy pomôže pri základnej orientácii, ktorý algoritmus sa hodí na aký typ úloh. Možno bude užitočnejší pohľad z druhej strany – vymenujeme typické úlohy a k nim priradíme vhodné algoritmy. Nesmie vás prekvapiť, že na väčšinu typov úloh je možné použiť viac druhov algoritmov.

### Klasifikácia

Do tejto skupiny úloh patrí hľadanie odpovedí na otázky typu „Aké typy kampaní a členských kariet má ponúkať moja obchodná firma?“ Aký okruh záujemcov osloví reklamná a marketingová kampaň? Prečo strácame zákazníkov? Bude zamýšľaný produkt úspešný?

### Najvhodnejšie algoritmy

Rozhodovacie stromy  
Naive Bayes  
Neurónové siete

### Použiteľné algoritmy

Zhlukovanie  
Sekvenčné zhlukovanie  
Asociačné pravidlá

**Regresia**

Zo štatistického hľadiska ide o metódu, ktorá kvantifikuje závislosť medzi dvoma spojitémi premennými: závislou premennou, ktorú sa snažíme predikovať a nezávislou teda prediktívnou premennou. Hľadáme priamku prechádzajúcu medzi jednotlivými bodmi, pre ktorú platí, že súčet druhých mocnín odchýlok od každého bodu je minimálna. Ak je vzťah medzi závislou a nezávislou premennou nelineárny, spravidla je potrebné transformovať prediktívnu premennú tak, aby umožnila nájsť lepšie preloženie. Logistická regresia je veľmi podobná lineárnej regresii, s tým rozdielom, že závislá premenná nie je spojitá, ale diskretná. Preto ju transformujeme na spojitú hodnotu, ktorá je funkciou pravdepodobnosti výskytu udalosti. Regresiu je možné použiť k predikovaní výsledkov dvoch, alebo viacerých úrovní, napríklad prečo nastane jav nesplácania pôžičiek, akú tržbu môžeme očakávať od ktorého zákazníka a podobne.

**Najvhodnejšie algoritmy**

Rozhodovacie stromy  
Neurónové siete

**Použiteľné algoritmy**

Zhlukovanie  
Sekvenčné zhlukovanie  
Naïve Bayes

**Segmentácia**

Rozdelenie nejakej množiny entít, napríklad zákazníkov do rôznych skupín segmentov podľa určitých kritérií umožňuje dosiahnuť cieľový a personalizovaný prístup ku každému segmentu. Iným okruhom využitia segmentácie je organizácia údajov, aby boli čo najefektívnejšie použiteľné. Pomocou segmentácie môžeme taktiež vyhľadávať rôzne chyby, pretože sú spravidla od ostatných prípadov vzdialené

**Najvhodnejšie algoritmy**

Zhlukovanie  
Sekvenčné zhlukovanie

**Použiteľné algoritmy**

Neurónové siete

**Analýzy textu**

Textové záznamy sú typickým príkladom neštruktúrovaných údajov, aj keď v nich určité pravidlá dokážeme identifikovať týkajú sa skôr syntaxe a sémantiky prirodzeného jazyka (čeština, angličtina...) v ktorom bol text vytvorený. Niektoré heuristické data miningové algoritmy si však poradia aj s bežným neštruktúrovaným textom

Použitie data miningových algoritmov pre niektoré typické druhy úloh je prehľadne usporiadané tabuľke

	Decision Trees	Naïve Bayes	Clustering	Seq. Clustering	Time Series	Association rules	Neural Network	
	✓	✓	✓	✓		✓	✓	Classification
	✓	✓	✓	✓				Regression
			✓	✓				Segmentation
	✓	✓	✓	✓		✓	✓	Assoc. Analysis
			✓	✓				Anomaly Detect.
				✓				Seq. Analysis
					✓			Time series

✓ - Najlepší    ✓ - Vhodný

Použitie algoritmov pre typické okruhy úloh

## Úvodný príklad pre data mining

Námetom pre prvý zoznamovací príklad z oblasti data miningu bude analýza tabuľky zákazníkov. Nakoľko databázy AdventureWorks a AdventureWorksDW sú firemné databázy firmy predávajúcej bicykle a ich príslušenstvo, budeme skúmať, či je zákazník potenciálnym kupcom bicykla. Využijeme algoritmy Microsoft Decision Tree a Microsoft Naive Bayes

Aby sme si uľahčili prácu pri návrhu data miningového modelu, najskôr budeme trochu pozornosti venovať príprave údajov.

Pomocou nástroja SQL Server Management Studio vytvoríme v databáze AdventureWorksDW tri množiny údajov – tri pohľady nad tabuľkami [dbo].[DimCustomer] a [dbo].[vDMPrep].

```
vDMCustomerTrain
vDMCustomerPredict
vDMCustomerValidate
```

Všetky tri pohľady v tomto najjednoduchšom prípade obsahujú rovnaké údaje. Spravidla sa v takýchto prípadoch rozdeľujú údaje do pohľadov podľa nejakého pseudonáhodného kľúča, napríklad ID modulo 3. Ak sa chcete viac priblížiť realite, skúste si vo vašom príklade údaje do jednotlivých pohľadov rozdeliť takto. Bude ich síce trikrát menej, ale každý pohľad bude obsahovať rôzne údaje. Prípadne môžete údaje rozdeliť len na dva pohľady, jeden na tréningovanie modelu a druhý na predikciu a overenie.

Prvú množinu použijeme pre „natréningovanie“ modelu. Druhá množina údajov bude slúžiť pre predikciu a tretia pre overenie predpovede

Column Name
CustomerKey
FirstName
LastName
Age
MaritalStatus
Gender
YearlyIncome
NumberChildrenAtHome
HouseOwner
NumberCarsOwned
EnglishCountryRegionName
Commute
BikeBuyer

*Návrhová štruktúra vytváraných pohľadov*

Nakoľko všetky tri pohľady majú rovnakú návrhovou štruktúru, uvádzame len skript pre vytvorenie jedného z nich

```
/****** Skript pre príklad data miningu *****/
```

```
USE [AdventureWorksDW]
go
```

```
/****** [dbo].[vDMCustomerPredict] *****/
CREATE VIEW [dbo].[vDMCustomerPredict] AS
SELECT c.[CustomerKey], c.[FirstName], c.[LastName],
       CASE
         WHEN Month(GetDate()) < Month(c.[BirthDate])
           THEN DateDiff(yy,c.[BirthDate],GetDate()) - 1
         WHEN Month(GetDate()) = Month(c.[BirthDate])
           AND Day(GetDate()) < Day(c.[BirthDate])
           THEN DateDiff(yy,c.[BirthDate],GetDate()) - 1
         ELSE DateDiff(yy,c.[BirthDate],GetDate())
       END AS [Age],
       c.[MaritalStatus], c.[Gender], c.[YearlyIncome], c.TotalChildren,
```

```

c.[NumberChildrenAtHome],
CASE c.[HouseOwnerFlag] WHEN 0 THEN ,No` ELSE ,Yes` END as HouseOwner,
c.[NumberCarsOwned], g.EnglishCountryRegionName,
c.[CommuteDistance] As Commute,
CASE x.[Bikes]
    WHEN 0 THEN ,No`
    ELSE ,Yes`
END AS [BikeBuyer]
FROM
    [dbo].[DimCustomer] c INNER JOIN (
        SELECT
            [CustomerKey] , [Region], [Age]
            , Sum( CASE [EnglishProductCategoryName]
                WHEN ,Bikes` THEN 1
                ELSE 0
            END) AS [Bikes]
        FROM
            [dbo].[vDMPrep]
        GROUP BY
            [CustomerKey] , [Region] , [Age]
        ) AS [x]
    ON c.[CustomerKey] = x.[CustomerKey]
join dimgeography g
on c.geographykey = g.geographykey
go

```

V skriptoch pre vytvorenie ďalších dvoch pohľadov je len zmenený názov

```

/***** [dbo].[vDMCustomerTrain] *****/
CREATE VIEW [dbo].[vDMCustomerTrain] AS
...

/***** View [dbo].[vDMCustomerValidate] *****/
CREATE VIEW [dbo].[vDMCustomerValidate] AS
...

```

Po ukončení precvičovania príkladu, prípadne ak chceme príklad opakovať je vhodné vytvorené pohľady z databázy AdventureWorksDW odstáť pomocou skriptu (uvádzame len skript pre odstránenie jedného pohľadu)

```

USE [AdventureWorksDW]
go

if exists (select * from dbo.sysobjects where id = object_id(N'[dbo].[vDMCustomerTrain]') and OBJECTPROPERTY(id, N'IsView') = 1)
DROP VIEW [dbo].[vDMCustomerTrain]

```

Pohľady obsahujú 18 484 záznamov. Príklad výpisu niekoľkých záznamov pohľadu uvádzame pre veľký počet stĺpcov rozdelený na dve časti

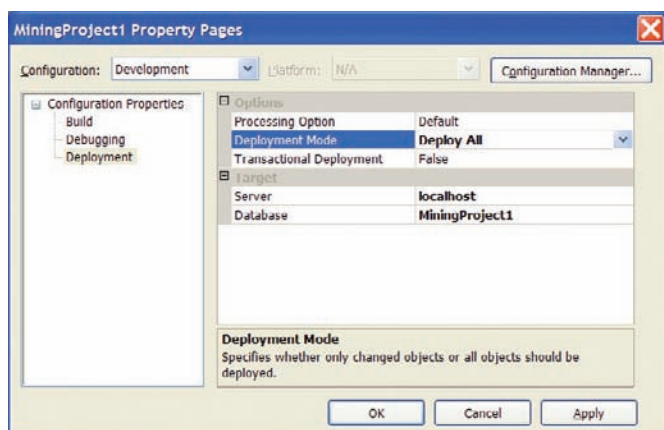
CustomerKey	FirstName	LastName	Age	MaritalStatus	Gender	YearlyIncome
11000	Jon	Yang	38	M	M	90000,00
11001	Eugene	Huang	39	S	M	60000,00
11002	Ruben	Torres	39	M	M	60000,00
11003	Christy	Zhu	36	S	F	70000,00
11004	Elizabeth	Johnson	36	S	F	80000,00
11005	Julio	Ruiz	39	S	M	70000,00
11006	Janet	Alvarez	39	S	F	70000,00

NumberChi	HouseOwner	NumberCarsOwned	EnglishCoun	Commute	BikeBuyer
0	Yes	0	Australia	1-2 Miles	Yes
3	No	1	Australia	0-1 Miles	Yes
3	Yes	1	Australia	2-5 Miles	Yes
0	No	1	Australia	5-10 Miles	Yes
5	Yes	4	Australia	1-2 Miles	Yes
0	Yes	1	Australia	5-10 Miles	Yes
0	Yes	1	Australia	5-10 Miles	Yes

V prostredí Visual Studio 2005 Studio vytvoríme nový projekt typu Analysis Services Project zo zložky BI projektov. V našom prípade sme zvolili názov MiningProject1. Všimnite si, že v zhode s filozofiou UDM vytvárame rovnaký typ analytického projektu ako pri vytváraní OLAP kociek. Rozdiel bude len v krokoch návrhu, ktoré sú dané záložkami. Vynecháme záložku Cubes a namiesto toho budeme vytvárať data miningový model. Vytváranie dátových zdrojov a pohľadov na dátové zdroje zostane nezmenené.

*Aj napriek tomu, že data mining sa oprávnenne považuje za vrchol databázových a analytických metód, jeho zvládnutie vďaka unifikácii na úrovni UDM modelovania nie je príliš zložitý, preto vás povzbudzujeme urobiť nasledujúci príklad aj v prípade, že problematike data miningu zatiaľ príliš nerozumieme. Príklad je možné urobiť takmer mechanicky podľa predpísaných krokov, a je takmer isté že na jeho konci budete mať o data miningu a jeho možnostiach oveľa jasnejšiu predstavu*

Nakoľko u skúsenejších čitateľov nepredpokladáme len mechanické prejdienie tohto cvičného príkladu, ale aj rôzne experimenty, je výhodné pre zrýchlenie deploymentu projektu na analytický server a zrýchlenie ladenia upraviť jeho spôsob. V okne Solutions Explorera cez kontextové menu aktivujeme dialóg Properties a v ňom zložku Deployment. V ňom nastavíme Deployment Mode na hodnotu Deploy All.



Nastavenie Deployment Mode

Podobne ako u OLAP analýz, aj v tomto prípade je postup budovania projektu naznačený poradím zložiek v okne vývojového prostredia Solution Explorer v pravom hornom rohu. Pre jednoduchý príklad data miningu budeme pracovať hlavne so záložkami.

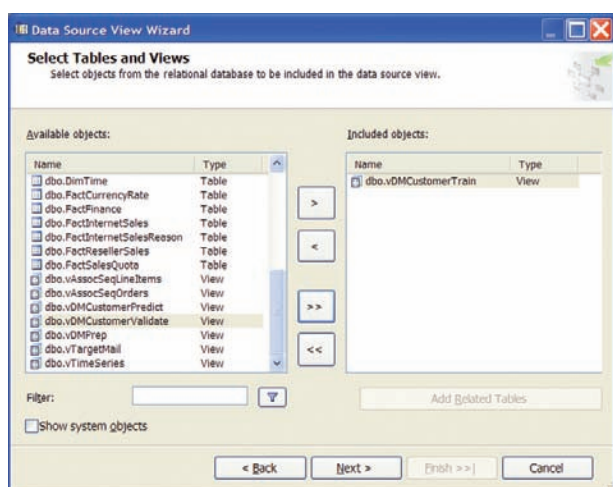
- Data Sources
- Data Source Views
- Mining Structures

### Definovanie dátových zdrojov

Ako dátový zdroj budú slúžiť nami vytvorené pohľady v databáze AdventureWorksDW obsahujúce vybrané údaje o zákazníkoch. Definujeme lokálny SQL server a databázu AdventureWorksDW. Pripojenie môže byť realizované napríklad cez Microsoft OLE DB Provider for SQL Server, alebo lepšie cez SqlClient Data Provider zo skupiny .Net Providers.

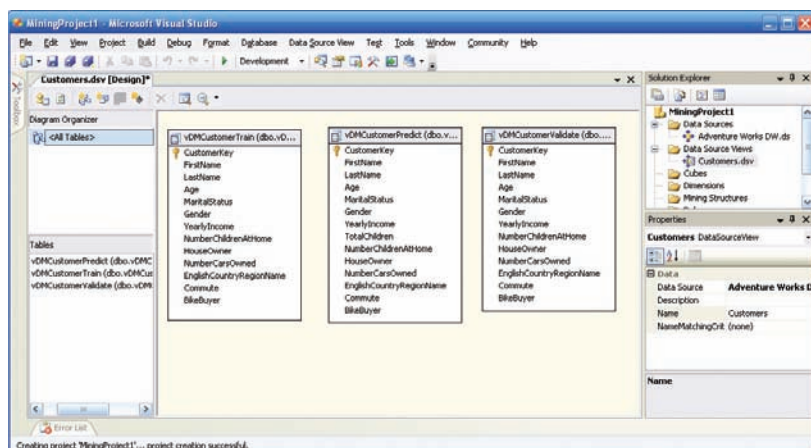
### Definovanie pohľadov na dátové zdroje

V tejto záložke vyberieme pohľady *dbo.vDMCustomerTrain* a *dbo.vDMCustomerPredict*. Pohľad môžeme nazvať napríklad Customers.



Definovanie pohľadov na dátové zdroje

V strednom okne vývojového prostredia sa zobrazí diagram návrhovej štruktúry vybraného pohľadu na dátové zdroje. V našom prípade je tvorený troma pohľadmi



Prezeranie údajov

Pomocou položky Explore Data kontextového menu si môžeme prezrieť vybrané údaje.

CustomerK	FirstName	LastName	Age	MaritalStat	Gender	YearlyInco	NumberChil	HouseOwn	NumberCar	EnglishCou	Commute	BikeBi
11000	Jon	Yang	38	M	M	90000	0	Yes	0	Australia	1-2 Miles	Yes
11001	Eugene	Huang	39	S	M	60000	3	No	1	Australia	0-1 Miles	Yes
11002	Ruben	Torres	39	M	M	60000	3	Yes	1	Australia	2-5 Miles	Yes
11003	Christy	Zhu	36	S	F	70000	0	No	1	Australia	5-10 Miles	Yes
11004	Elizabeth	Johnson	36	S	F	80000	5	Yes	4	Australia	1-2 Miles	Yes
11005	Julio	Ruz	39	S	M	70000	0	Yes	1	Australia	5-10 Miles	Yes
11006	Janet	Alvarez	39	S	F	70000	0	Yes	1	Australia	5-10 Miles	Yes
11007	Marco	Mehta	40	M	M	60000	3	Yes	2	Australia	0-1 Miles	Yes
11008	Rob	Verhoff	40	S	F	60000	4	Yes	3	Australia	10+ Miles	Yes
11009	Shannon	Carlson	40	S	M	70000	0	No	1	Australia	5-10 Miles	Yes
11010	Jacquelyn	Suarez	40	S	F	70000	0	No	1	Australia	5-10 Miles	Yes
11011	Curts	Lu	41	M	M	60000	4	Yes	4	Australia	10+ Miles	Yes
11012	Lauren	Walker	36	M	F	100000	0	Yes	2	United Stat	1-2 Miles	No
11013	Ian	Jenkins	36	M	M	100000	0	Yes	3	United Stat	0-1 Miles	No
11014	Sydney	Bennett	36	S	F	100000	0	No	3	United Stat	1-2 Miles	No
11015	Chloe	Young	25	S	F	30000	0	No	1	United Stat	5-10 Miles	Yes
11016	Wyatt	Hill	25	M	M	30000	0	Yes	1	United Stat	5-10 Miles	Yes

Prezeranie údajov

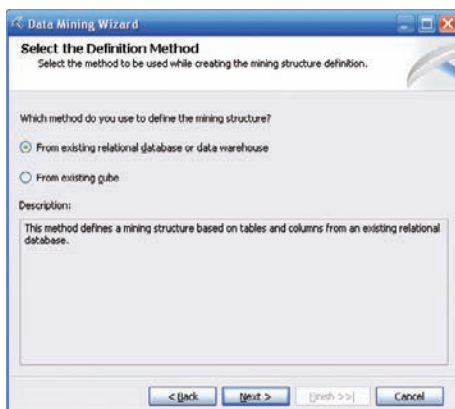
### Návrh data miningového modelu

Po prípravných fázach definovania zdroja údajov pristúpime k vytvoreniu data miningového modelu. V zložke Mining Structures aktivujeme sprievodcu Data Mining Wizard. Tento sprievodca umožní vytvoriť data miningový model buď z relačnej databázy, alebo z OLAP kocky. V tomto cvičnom príklade budeme model vytvárať z relačnej databázy.



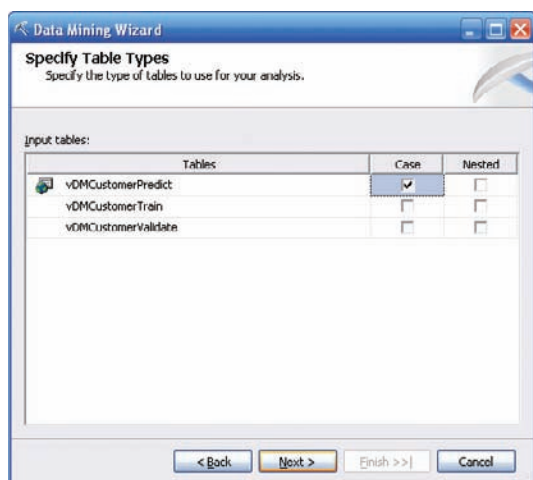
Data Mining Wizard

V dialógu pre výber štruktúry údajov, ktorú plánujeme podrobiť data miningu označíme voľbu From existing relational database or data warehouse



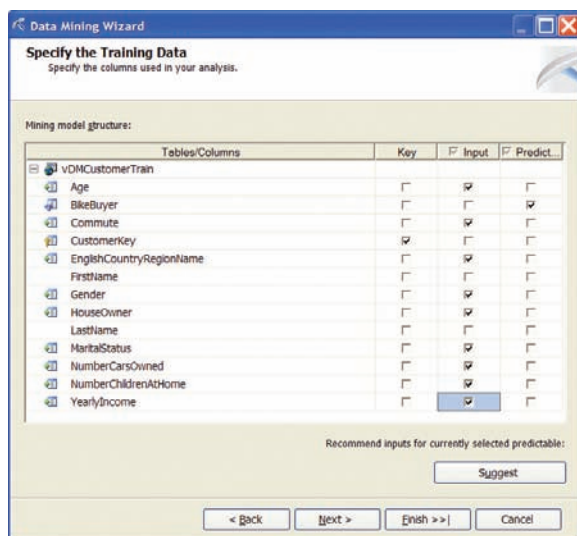
Data miningový model môžeme vytvárať z existujúcej databázy alebo dátového skladu, prípadne z existujúcej OLAP kocky

Skôr než začneme upresňovať údaje pre data miningový model, je potrebné určiť tabuľku s ktorou bude model pracovať, alebo laicky povedané – na ktorej sa bude učiť.



Výber tabuľky prípadov, na ktorých sa bude data miningový model „učiť“

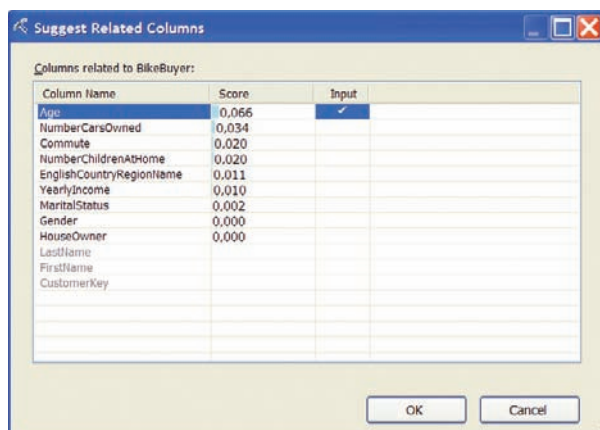
Nasleduje výber algoritmu. Najskôr vyskúšame nevyvážené rozhodovacie stromy (Microsoft Decision Tree), teda algoritmus, ktorý bol implementovaný aj v predchádzajúcej verzii SQL Serveru 2000. Po výbere zdroja údajov (Customers) a špecifikovaní tabuľky (pohľadu) sme sa prepracovali ku kľúčovému dialógu pre špecifikovanie modelu. Potrebujeme definovať najskôr kľúčový stĺpec pomocou ktorého bude jednoznačne identifikovaná každá entita. V našom prípade je to jednoznačne CustomerKey, nakoľko tento stĺpec je aj primárnym kľúčom zdrojovej tabuľky (pohľadu). Nasleduje špecifikácia vstupných stĺpcov a stĺpca ktorého hodnotu chceme predikovať. Predmetom nášho záujmu je, či ten ktorý zákazník si u nás kúpi bicykel, takže ako predikovateľný stĺpec označíme BikeBuyer. Ostatné stĺpce s výnimkou tých od ktorých potenciálna kúpyschopnosť bicykla vôbec nezávisí, teda mena a priezviska označíme ako vstupné. Môžeme si to dovoliť, nakoľko sme si vytvorením pohľadu, ktorý obsahuje len predmetné údaje značne zjednodušili prácu a pravdepodobne aj zvýšili rýchlosť vykonávania algoritmu.



Odporúčanie výberu stĺpcov, od ktorých závisí výsledok analýzy

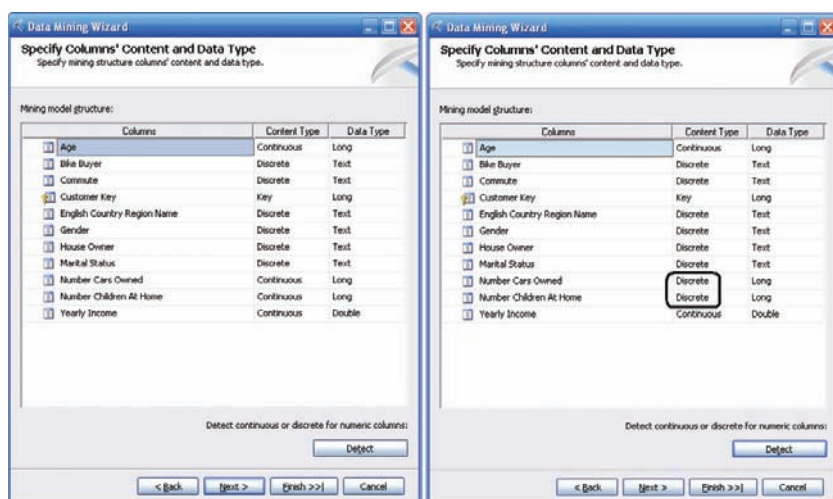
Pri kvalifikovanom výbere vstupných stĺpcov nám môže pomôcť odporúčenie sprievodcu, ktoré je dostupné tlačidlom Suggest.





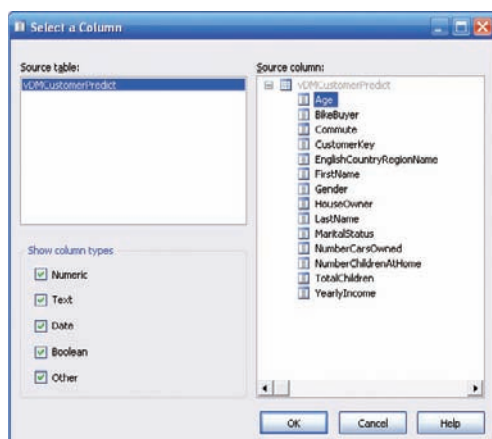
Odporúčanie výberu stĺpcov, od ktorých závisí výsledok analýzy

Po výbere vstupných stĺpcov ešte môžeme špecifikovať, či obsahujú diskkrétne, alebo spojité hodnoty. Niektoré hodnoty sú typicky spojité, napríklad vek (čas plynie spojitou), sumy obeživa a podobne, iné stĺpce obsahujú diskkrétne údaje. Typickou diskkrétou hodnotou je napríklad počet detí.



Detekcia diskrétnych a spojitých atribútov, vľavo situácia pred autodetekciou, vpravo po autodetekcii

Pre algoritmus Microsoft Naive Bayes však budeme potrebovať aj vek ako diskkrétne hodnotu, nakoľko tento algoritmus so spojitémi hodnotami nepracuje. To dosiahneme pridaním stĺpca, prostredníctvom menu Columns || Add a Column v stromovej štruktúre okna Mining Structure v ľavom zvislom stĺpci obrazovky vývojového prostredia. Stĺpec bude obsahovať údaje z pôvodného stĺpca Age, preto ho nazveme výstižne napríklad AgeDiscretized.



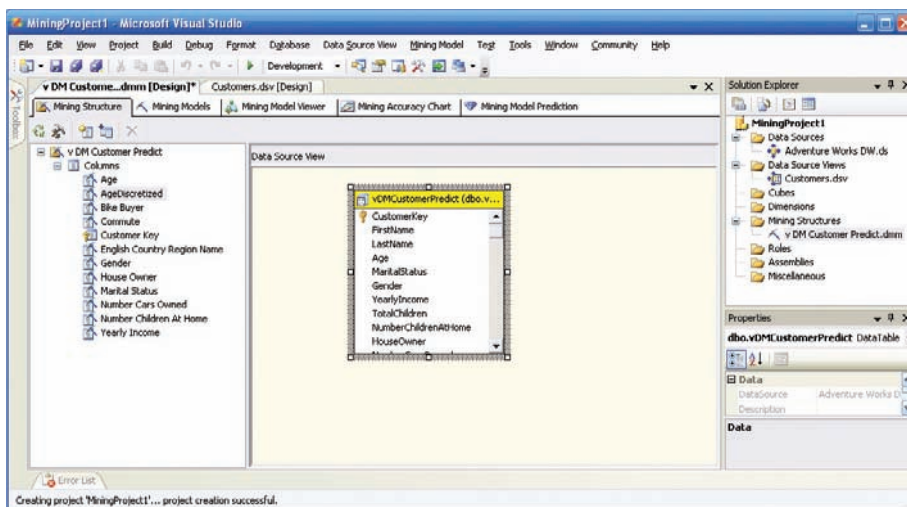
Výber stĺpca „Age“ ako zdrojového stĺpca pre stĺpec obsahujúci diskkrétne hodnoty veku

Počas vytvárania nového stĺpca budeme upozornení, že stĺpec Age už bol zahrnutý do návrhu. Tlačidlom Yes toto varovanie zoberieme na vedomie. Vo vlastnostiach tohto stĺpca nastavíme parameter Content na hodnotu Discretized. Pre nevyvážené rozpadové stromy budeme túto hodnotu ignorovať.

Všimnime si rozloženie pracovnej obrazovky. Hlavné okno je rozložené na záložky.

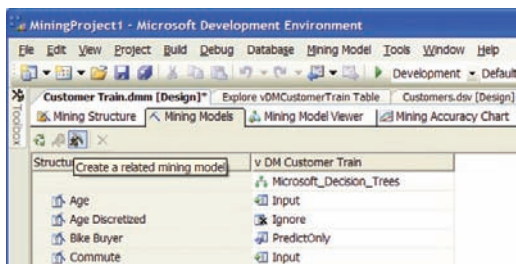
- Mining Structure
- Mining Models
- Mining Model Viewer
- Mining Accuracy Chart
- Mining Model Prediction

Záložka **Mining Structure** obsahuje zoznam stĺpcov.

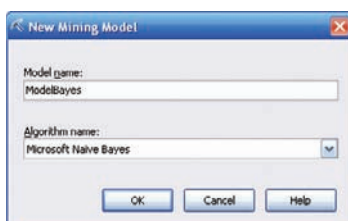


Data miningový model v prostredí vývojového prostredia (záložka Mining Structure)

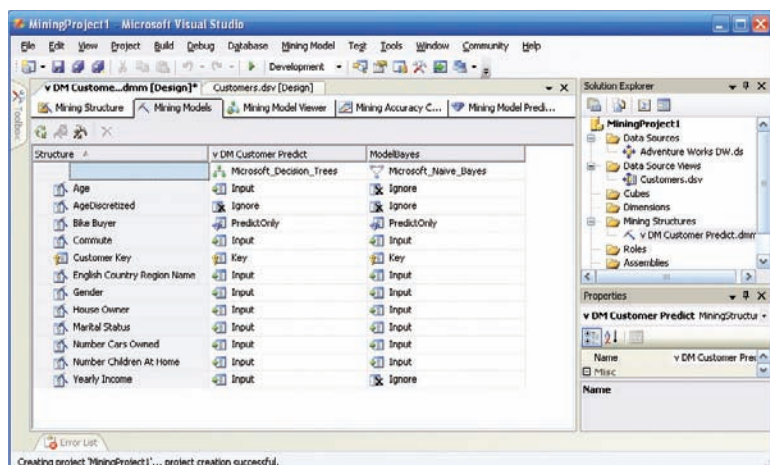
Momentálne nás bude najviac zaujímať záložka **Mining Models**, kde sú vybrané vstupné stĺpce, predikovaný a kľúčový stĺpec pre každý data miningový model. Zatiaľ v tomto príklade máme vytvorený len jeden. Z neho však ľahko odvodíme (tlačidlo Create Related Mining Model toolbaru v tejto záložke) ďalší model, v našom prípade Microsoft Naive Bayes. Tento algoritmus bude pracovať len s diskretnými hodnotami. Stĺpce obsahujúce spojitú hodnotu bude jednoducho ignorovať.



Vytvorenie závislého modelu



Dialóg pre vytvorenie závislého modelu



Dva závislé data miningové modely v prostredí BI Dev Studio

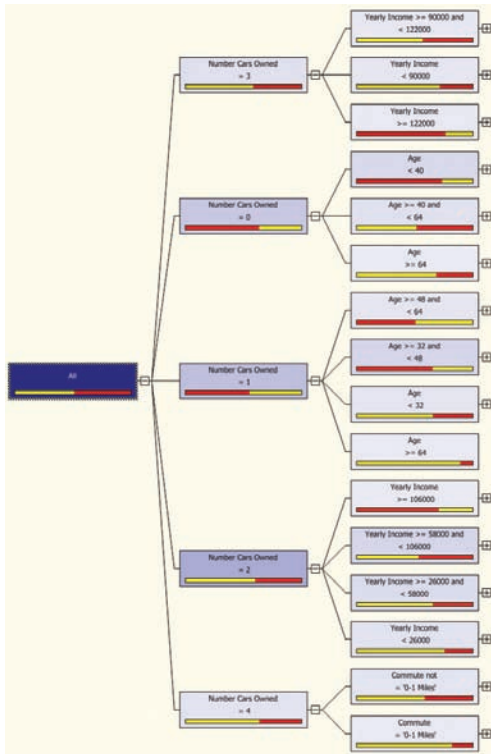
Pomaly ale isto sa blížíme k cieľu. Po nastavení všetkých parametrov môžeme data miningový model zostaviť a zaviesť na server (menu Build | Deploy Solution). Postup zostavovania môžeme sledovať vo výstupnom okne

```
----- Build started: Project: MiningProject1, Configuration: Development -----
Started Building Analysis Services project: Incremental ....
Build complete -- 0 errors, 0 warnings
----- Deploy started: Project: MiningProject1, Configuration: Development -----
Performinga full deployment of the ,MiningProject1\` database to the ,localhost\`
server.
Generating deployment script...
    Add Database MiningProject1
    Process MiningStructure v DM Customer Predict
Done
Sending deployment script to the server...
Done
Deploy complete -- 0 errors, 0 warnings
===== Build: 1 succeeded or up-to-date, 0 failed, 0 skipped =====
===== Deploy: 1 succeeded, 0 failed, 0 skipped =====
```

### Mining Model Viewer

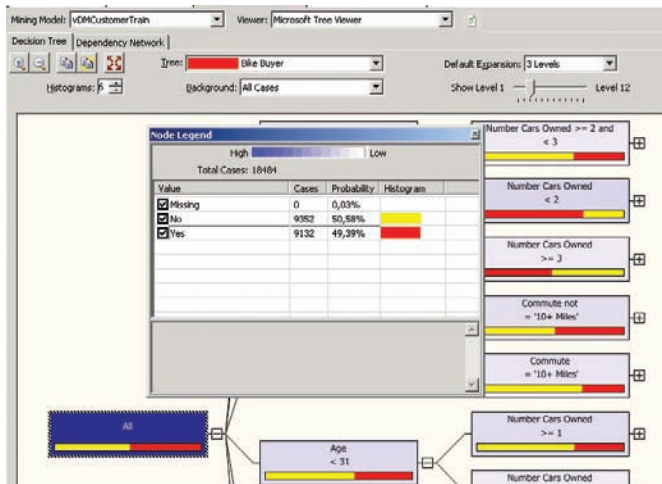
Táto záložka je riešená univerzálne pre prezeranie výsledkov činnosti všetkých algoritmov. V našom prípade sme navrhli model pre dva algoritmy, takže aj v okne Model Viewer si môžeme prezerat' Po zostavení modelu sa dej prenesie do záložky Mining Model Viewer. Na zobrazenom diagrame nevyváženého rozpadového stromu môžeme analyzovať hĺbku jednotlivých závislostí a vzťahy vo vnútri nájdenej hierarchie. Čím je farba políčka tmavšia, tým je pravdepodobnosť výskytu tejto vlastnosti vyššia. Každý obdĺžnik diagramu stromovej štruktúry obsahuje aj pomerový indikátor.

Pomocou symbolov +a – môžeme rozbaľovať, prípadne zbaľovať ďalšie úrovne hierarchie.



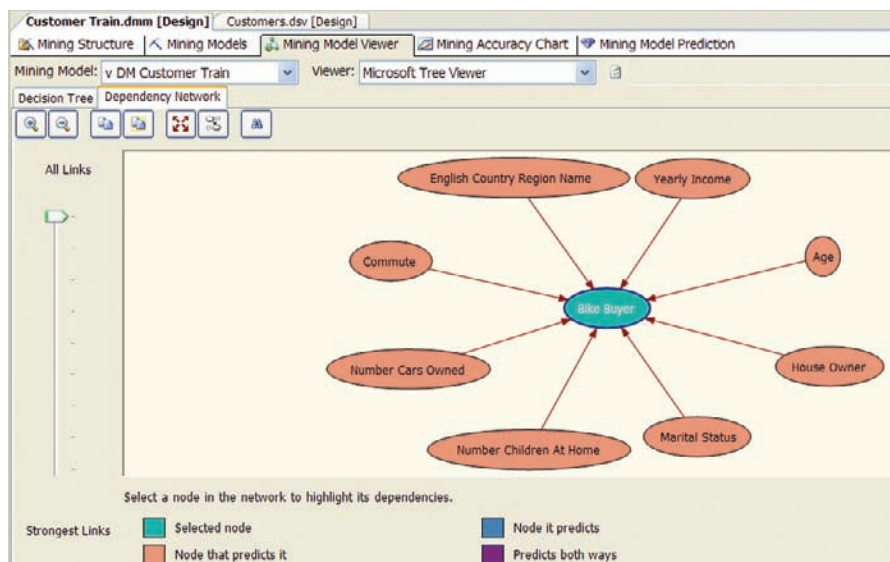
Decision Tree

Pomer závislosti výskytu predikovanej veličiny vieme približne určiť z pomerového stĺpcového diagramu. V našom prípade žltá farba identifikuje počet zákazníkov, ktorí si bicykel nekúpili a naopak červená farba znázorňuje podiel zákazníkov, ktorí si bicykel kúpili. Kliknutím na konkrétny uzol zobrazíme aj presné percentuálne podiely v okne Node Legend. Štandardne je toto okno umiestnené v pravom hornom rohu obrazovky.



Legenda uzlov

Práca s grafom závislostí (Dependency Network) je úplne intuitívna. Vo forme prehľadného diagramu zobrazí od čoho a v akej miere závisí predikovateľná veličina. Posúvaním „potenciometra“ v ľavej časti pracovnej obrazovky môžeme slabšie väzby postupne eliminovať, takže na konci nám zostanú len najsilnejšie závislosti.



Dependency Network

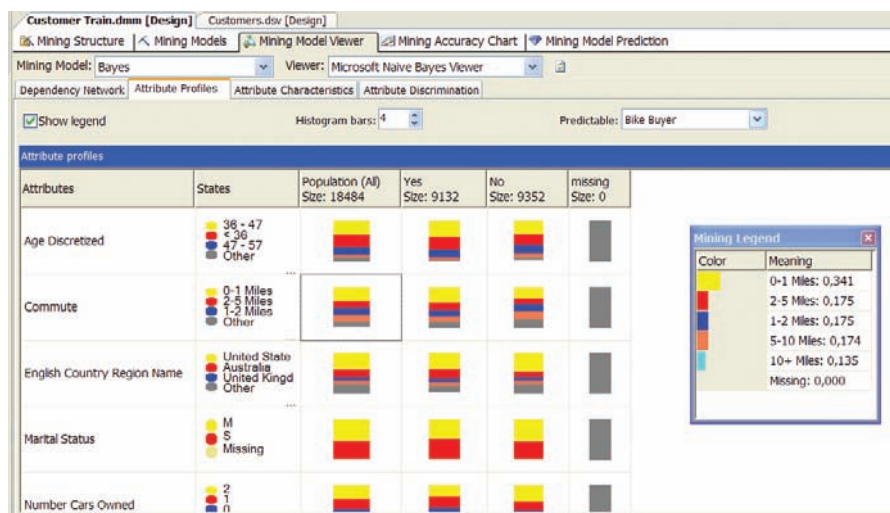
### Algoritmus Microsoft Naive Bayes

V prípade prezerania si výsledkov analýz tohto algoritmu bude pracovná obrazovka rozdelená na štyri ďalšie záložky

- Dependency Network
- Attribute Profiles
- Attribute Characteristic
- Attribute Discrimination

Práca s grafom závislostí (Dependency Network) je rovnaká ako pri nevyvážených rozpadových stromoch.

V ostatných záložkách sú v prehľadnej grafickej podobe zobrazené výsledky analýz pre jednotlivé testované atribúty závislosti.



Profil atribútov

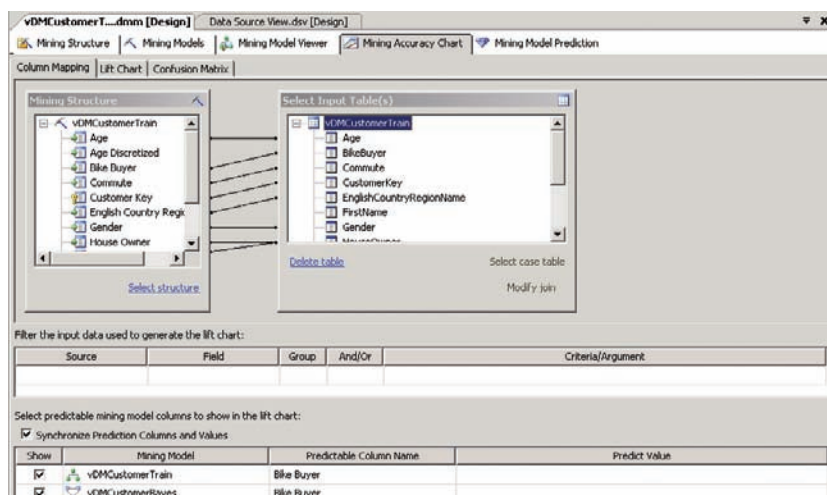


Rozdelenie pravdepodobnosti

### Mining Accuracy Chart

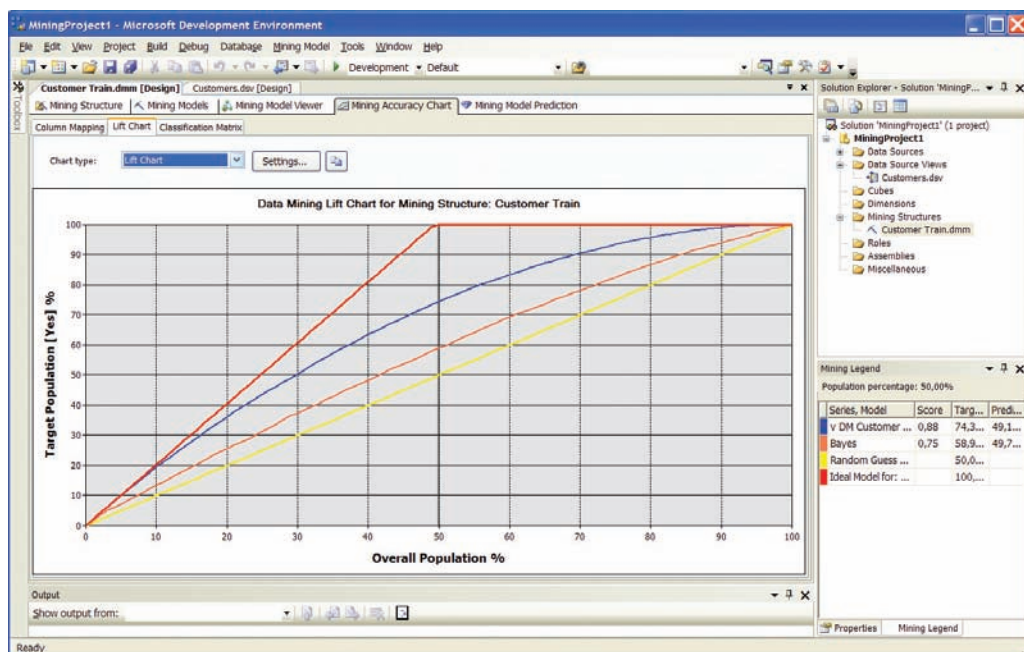
Výhodou novej verzie SQL Serveru je možnosť zobrazovania prehľadných Lift Chart grafov pre testovanie data miningových modelov. Princíp grafu je jednoduchý. Najskôr zoradíme výsledky podľa klesajúcej pravdepodobnosti predpovedia porovnáme ich s náhodným výberom (najhorší možný prípad) a s najlepším teoretickým výberom. Krivka pre správne navrhnutý model by samozrejme mala byť čo najbližšie ideálnej krivke a jej sklon by sa mal plynulo znižovať. V opačnom prípade máme nesprávne zvolený model.

Pre získanie grafu závislosti je potrebné najskôr nadefinovať vzťahy medzi dataminingovým modelom a štruktúrou tabuľky vstupných údajov.



Charakteristiky atribútov

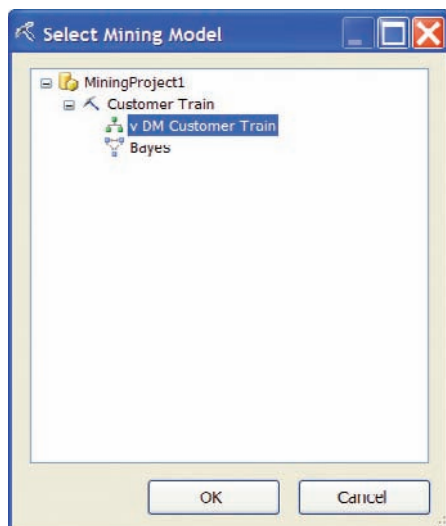
Na obrázku vidíme, že najlepším algoritmom pre túto úlohu sú nevyvážené rozpadové stromy. Algoritmus Naive Bayes je oveľa horší a skoro sa blíži náhodnému výberu



Lifting Chart

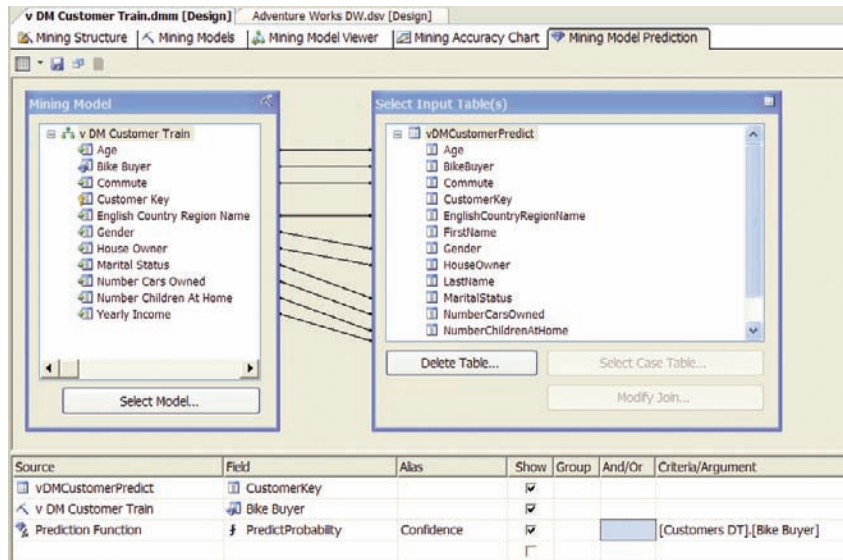
### Predikcia

Návrh dotazu pre predikciu sa bude odohrávať v záložke Mining Model Prediction. Najskôr zvolíme typ algoritmu, ktorý pre predikciu plánujeme použiť. Ktorý algoritmus z dvojice použitých je lepší nám už prezradil Lift chart v záložke Mining Accuracy Chart.



Výber modelu pre predikciu

Ako zdroj údajov pre predikciu použijeme pohľad vDMCustomerPredict. Na obrazovke sa zobrazia dve tabuľky, v jednej sú stĺpce z modelu a v druhej stĺpce z predikčnej tabuľky. Pre návrh predikčného dotazu slúži dolná časť obrazovky, kam môžeme presúvať polia z modelu, prípadne predikčnej tabuľky. Z predikčnej tabuľky ako prvú hodnotu presunieme kľúčový stĺpec, ktorý popisuje jednotlivé prípady, teda CustomerKey. Z modelu použijeme stĺpec Bike Buyer, hodnotu ktorého chceme predikovať. V treťom riadku nastavíme Prediction Function, field PredictProbability. Do stĺpca Kriteria /Argument zadáme hodnotu [Customers DT].[Bike Buyer]. Situáciu názorne popisuje obrázok.



*Výber tabuľky a priradenie stĺpcov pre predikciu*

O tom koľko práce sme vlastne v spolupráci s wizardom pre návrh predikčného dotazu vykonali interaktívnym vyplnením troch riadkov tabuľky sa môžeme presvedčiť ak si necháme novovytvorený predikčný dotaz zobrazit

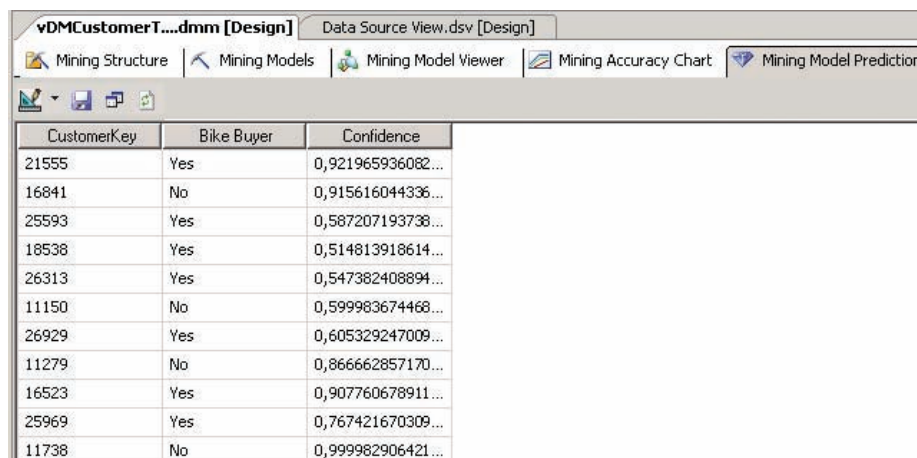
```

SELECT
    t.[CustomerKey],
    [v DM Customer Train].[Bike Buyer],
    (PredictProbability([Customers DT].[Bike Buyer])) as [Confidence]
From
    [v DM Customer Train]
PREDICTION JOIN
    OPENQUERY([Adventure Works DW],
        ,SELECT
            [CustomerKey],
            [Age],
            [MaritalStatus],
            [Gender],
            [YearlyIncome],
            [NumberChildrenAtHome],
            [HouseOwner],
            [NumberCarsOwned],
            [EnglishCountryRegionName],
            [Commute],
            [BikeBuyer]
        FROM
            [dbo].[vDMCustomerPredict]
        ,) AS t
ON
    [v DM Customer Train].[Age] = t.[Age] AND
    [v DM Customer Train].[Marital Status] = t.[MaritalStatus] AND
    [v DM Customer Train].[Gender] = t.[Gender] AND
    [v DM Customer Train].[Yearly Income] = t.[YearlyIncome] AND
    [v DM Customer Train].[Number Children At Home] = t.[NumberChildrenAtHome] AND
    [v DM Customer Train].[House Owner] = t.[HouseOwner] AND
    [v DM Customer Train].[Number Cars Owned] = t.[NumberCarsOwned] AND
    [v DM Customer Train].[English Country Region Name] = t.[EnglishCountryRegionName]
AND
    [v DM Customer Train].[Commute] = t.[Commute] AND
    [v DM Customer Train].[Bike Buyer] = t.[BikeBuyer]

```



Po prepnutí hlavného okna do režimu Results si môžeme prezrieť výsledky predikcie. Pre každý konkrétny prípad je v tabuľke jednak výsledok predikcie a taktiež aj koeficient zhody, čím si môžeme overiť hodnovernosť predikcie každého konkrétneho prípadu.



CustomerKey	Bike Buyer	Confidence
21555	Yes	0,921965936082...
16841	No	0,915616044336...
25593	Yes	0,587207193738...
18538	Yes	0,514813918614...
26313	Yes	0,547382408894...
11150	No	0,599983674468...
26929	Yes	0,605329247009...
11279	No	0,866662857170...
16523	Yes	0,907760678911...
25969	Yes	0,767421670309...
11738	No	0,999982906421...

*Výsledky predikcie*

## Kapitola 6: Reportovacie služby

Ak si pozrieme obrázok pyramídy v úvodnej kapitole, zistíme, že 65 až 80 % bežných používateľov BI služieb vyžaduje informácie spracované vo forme reportov. Reporty by mali slúžiť pre podporu rozhodovania na všetkých stupňoch organizačnej infraštruktúry. Report však nemôžeme chápať len ako bežný dokument, má svoje špecifiká a hlavne životný cyklus. Návrhom reportu, nech by bol hocikako kvalitný a trebárs aj nadčasový a zohľadňoval by všetky požiadavky sa celý proces reportovania nekončí. Rovnako dôležitá je aj jeho správa a taktiež spôsob prístupu – teda ako sa k reportom dostanú jeho adresáti. Ako vyplýva z názvu sú to služby integrované do SQL Serveru 2005, ktoré sa nasadzujú 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 prispôbiť tak, aby sme jednoducho 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 oblastia 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 prístupujú 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.

## Životný cyklus reportu

Životný cyklus reportu začína jeho návrhom, teda vytvorením predpisu, 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 odsek, dospeli by sme k trom fázam životného cyklu reportu

- Návrh
- Správa
- Doručenie

Medzi poslednými dvomi etapami, teda správou reportu a procesom jeho doručovania je obojsmerná interakcia.



Životný cyklus reportovania

### Návrh reportu

V tejto etape vývojári navrhujú reporty, ktoré budú následne publikované prostredníctvom Report Servera. Pre vývoj reportu je možné využiť nástroj Microsoftu – Visual Studio, 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

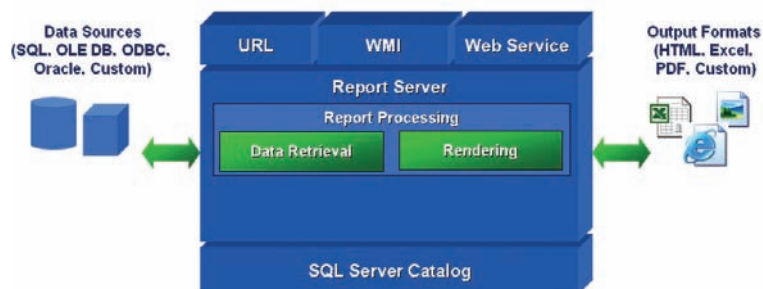
### Správa reportu

V tejto etape spravujeme a zladujeme 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.

### Architektúra reportovacích služieb

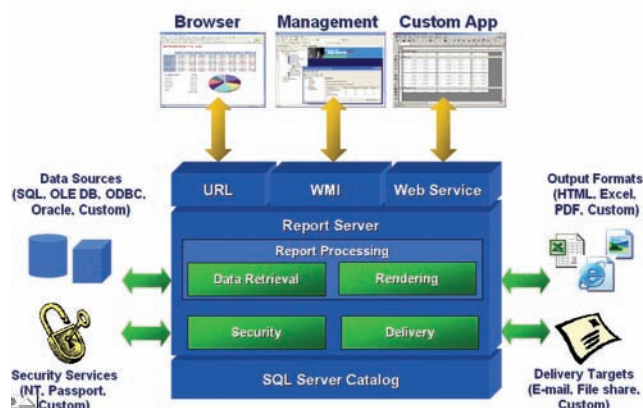
Architektúra reportovacích služieb je niekoľkovrstvová. Základnú vrstvu tvorí SQL Server Catalog, čo je databáza pod správou SQL Servera. Údaje z tejto databázy využíva Report Server pre ukladanie metadát, snapshotov, definícií reportov, adresárov, údajov pre zabezpečenie a podobne. Dôležitou informáciou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

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



Procesy na úrovni reportovacieho servera

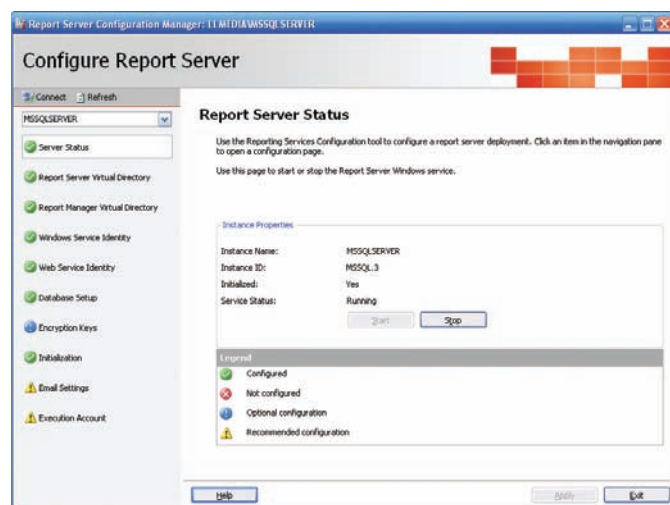
Na primárne aplikačné rozhrania sú naviazané aplikácie. Na URL rozhranie pristupujeme pomocou prehliadača webového obsahu, napríklad Internet Explorera. Cez WMI rozhranie pristupujeme z administrátorskej konzoly. Bežné aplikácie využívajú rozhranie webovej služby



Komplexný diagram architektúry reportovacích služieb a ich najbližšieho IT okolia

## Nástroj pre konfiguráciu reportovacích služieb

Nástroj pre konfiguráciu reportovacích služieb je prístupný aj cez menu operačného systému *Start – Programs – Microsoft SQL Server – Configuration tools – Report Services Configuration*. Prostredníctvom tohto vizuálneho nástroja dokážeme získať a v prípade potreby aj zmeniť väčšinu dôležitých parametrov pre prácu reportovacích služieb



Nástroj pre konfiguráciu reportovacích služieb

## Komplexný projekt

Účelom príkladu je nielen zoznámenie sa s vývojovým prostredím v režime návrhu reportu, ale hlavne kompletný postup vytvorenia jednoduchého reportu.

### 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 je vytvorený nad cvičnou databázou AdventureWorks.

Report bude obsahovať atribúty *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

<b>Region</b>	SalesTerritory.Name
<b>Category</b>	ProductCategory.Name
<b>SubCategory</b>	ProductSubCategory.Name

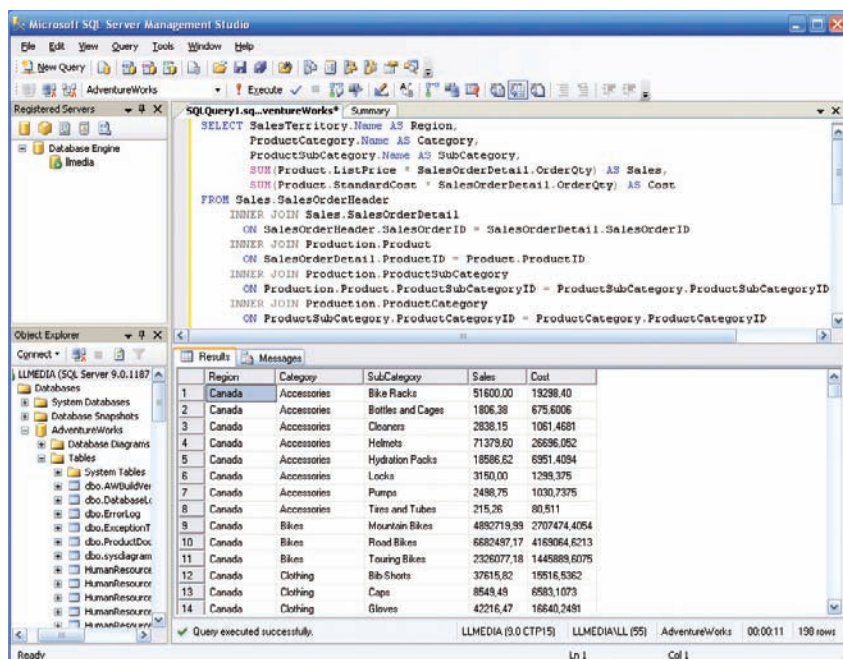
Štvrtý a piaty stĺpec sú výsledkom agregáčnych funkcií SUM

<b>Sales</b>	SUM(Product.ListPrice * SalesOrderDetail.OrderQty)
<b>Cost</b>	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 Sales.SalesOrderHeader
     INNER JOIN Sales.SalesOrderDetail
       ON SalesOrderHeader.SalesOrderID = SalesOrderDetail.SalesOrderID
     INNER JOIN Production.Product
       ON SalesOrderDetail.ProductID = Product.ProductID
     INNER JOIN Production.ProductSubCategory
       ON Production.Product.ProductSubCategoryID = ProductSubCategory.
ProductSubCategoryID
     INNER JOIN Production.ProductCategory
       ON ProductSubCategory.ProductCategoryID = ProductCategory.ProductCategoryID
     INNER JOIN Sales.SalesPerson
       ON Sales.SalesPerson.SalesPersonID = SalesOrderHeader.SalesPersonID
     INNER JOIN Sales.SalesTerritory
       ON SalesPerson.TerritoryID = SalesTerritory.TerritoryID
     INNER JOIN Sales.Customer
       ON SalesOrderHeader.CustomerID = Customer.CustomerID
WHERE SalesTerritory.[Group] = ,North America`
     AND Customer.CustomerType = ,S`
GROUP BY SalesTerritory.Name, ProductCategory.Name, ProductSubCategory.Name
```

Dopyt vyskúšame, prípadne odladíme pomocou konzolovej aplikácie, napríklad pomocou SQL Server Management Studia. SQL dopyt môžeme vyskúšať a odladiť aj vo fáze návrhu pomocou nástroja Query Builder, no pre ladenie zložitejších dopytov nám Management Studio poskytuje predsa len lepšie možnosti. Pomerne dôležitým faktorom je aj počet záznamov vo výpise. Ak ich je niekoľko stovák tisíc tak v taký report nemá pre nikoho žiadnu cenu, pretože neposkytne nijaký prehľad. V našom prípade má výsledný výpis 198 riadkov, čo je veľmi prijateľná hodnota, pretože report v tlačenej podobe bude mať najviac štyri až päť strán



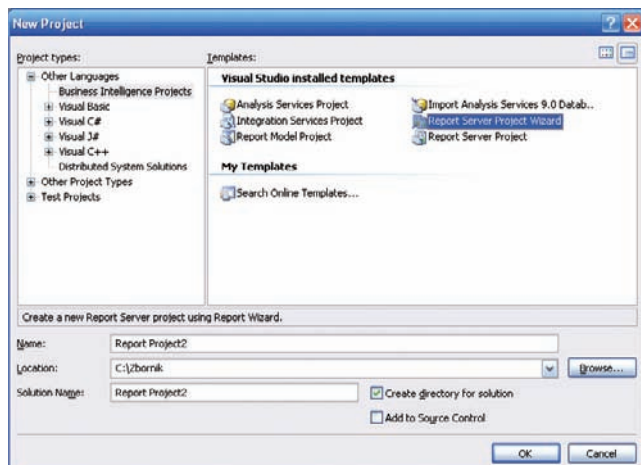
Ladenie SQL dopytu pomocou SQL Server Management Studia

### Návrh reportu

Teraz už môžeme prísť k vytvoreniu projektu. V hlavnom menu Visual Studia aktivujeme položku menu pre vytvorenie nového projektu. V zložke typov projektov Business Intelligence Projects máme k dispozícii tri šablóny projektov

- Report Model Project
- Report Server Project Wizard
- Report Server Project

Pre túto aplikáciu využijeme šablónu Report Project Wizard a nový projekt nejakú nazveme.



Dialóg pre vytvorenie nového projektu vo vývojovom prostredí Visual Studia

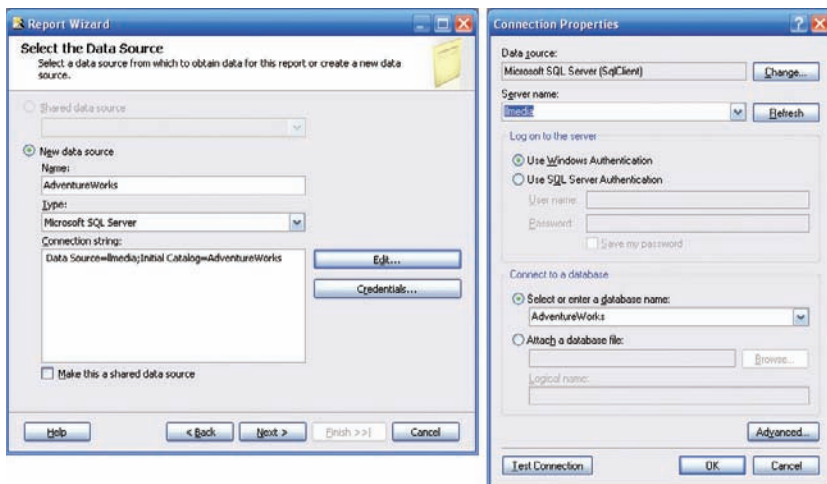
Po zadaní názvu projektu sa aktivuje sprievodca vytvorením reportu. Sprievodca nám bude nápomocný v nasledujúcich fázach návrhu

- 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



Úvodný dialóg sprievodcu pre vytvorenie reportu

Prvým krokom sprievodcu je výber dátového zdroja a nastavenie parametrov pre pripojenie sa k databáze



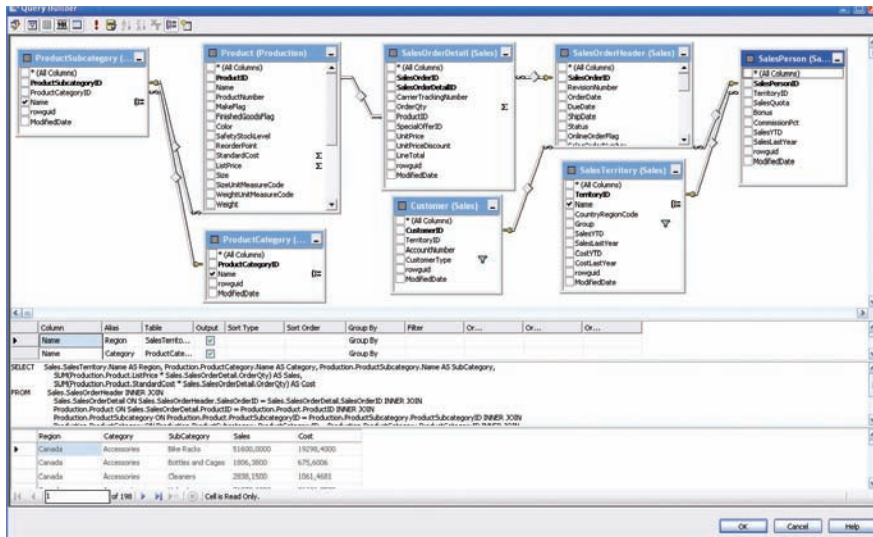
Definovanie databázy AdventureWorks ako zdroja údajov

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“



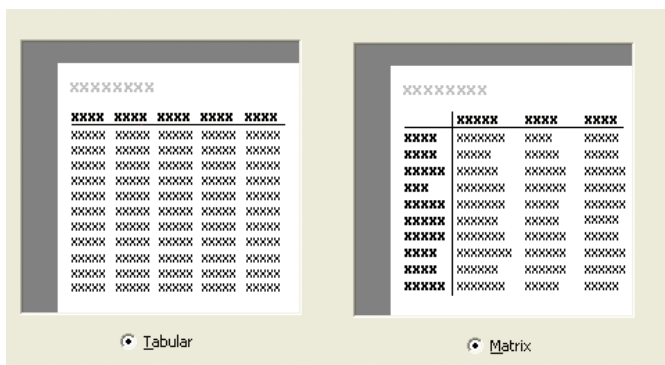
Dialóg pre návrh dopytu slúžiaceho k výberu údajov

V tejto etape môžeme využiť aj nástroj Query Builder, kde môžeme SQL dopyt navrhnuť aj graficky.



Návrh a otestovanie dopytu pomocou nástroja Query Builder

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



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

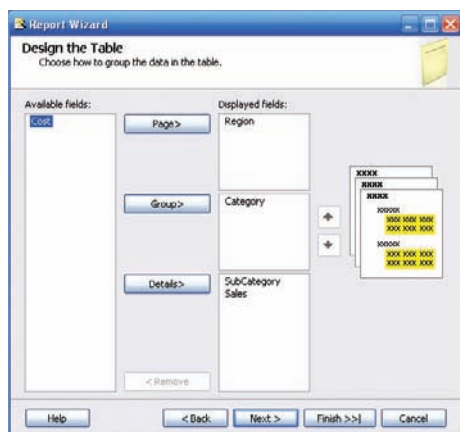
- Page
- Group
- Details

Pri tejto činnosti si musíme predstaviť, ako bude nami navrhovaný report vyzeráť 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 objavajú.

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

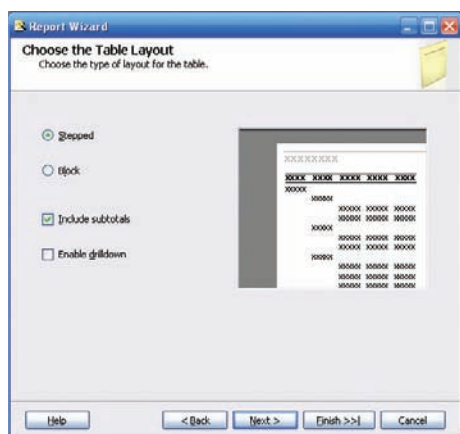
<b>Page</b>	Region
<b>Group</b>	Category
<b>Details</b>	Subcategory Sales





Dialóg pre návrh štruktúry tabuľky

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



Dialóg „Choose the Table Layout“

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



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

Na výber sú ponúkané možnosti

- Bold
- Casual
- Corporate
- Compact
- Plain

Jednotlivé možnosti opäť najlepšie predstavi ilustrачný obrázok. My sme zvolili štýl Corporate

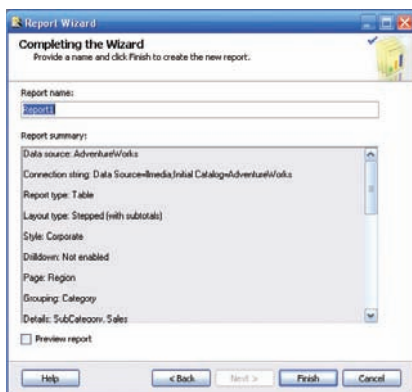


Ilustračná schéma výberu štýlu dokumentu

Týmto krokom sme vlastne návrh reportu zavřili. Nasleduje už len dialóg pre nastavenie parametrov deploymentu, ktorý umožňuje zadať prípadne zmeniť dva parametre - URL adresu reportovacieho servera a adresár projektu

Report Server: <http://localhost/ReportServer>

Deployment Folder: Report Project2



Záverečný dialóg sprievodcu vytvorením reportu

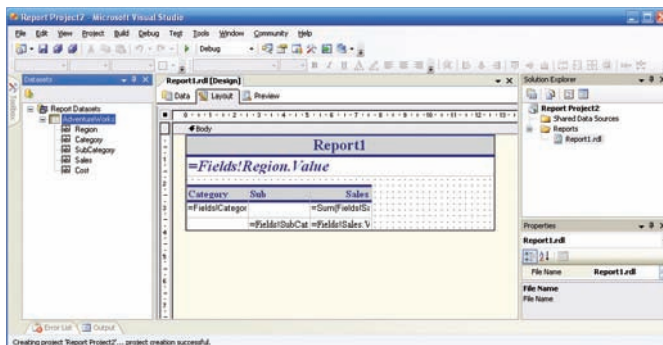
V záverečnom dialógu si môžeme pozrieť sumárne informácie o tomto reporte. Report môžeme vhodne pomenovať. Na záver tejto fázy návrhu sa radšej trochu poistíme. V menu vývojového prostredia File aktivujeme možnosť Save All.

## Práca s reportom v prostredí Visual Studio

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 plochy. Túto plochu je možné prepnúť do režimov

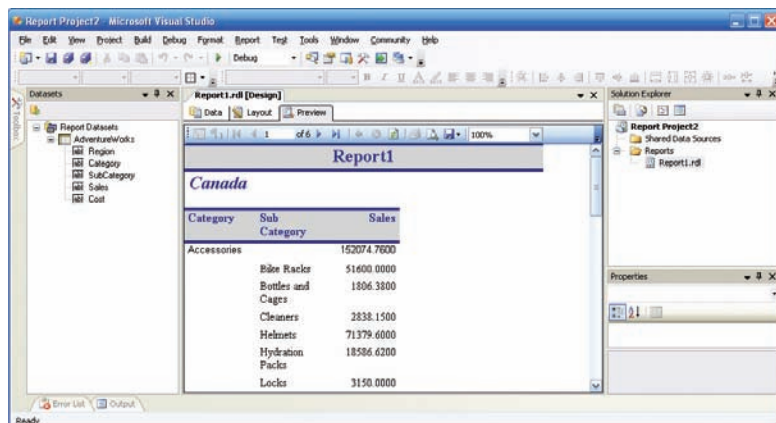
- **Data** pre pripojenie sa k databáze a návrh SQL dopytu
- **Layout** pre návrh formulára reportu
- **Preview** pre prehľadanie reportu.

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



Visual Studio – pracovná plocha vývojového prostredia po vytvorení nového projektu – režim Layout

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 prispôbení 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 vývojového prostredia sú ikony pre prepínanie jeho módov. Môžeme prepínať medzi návrhovým zobrazením, LAYOUT to je stav, v ktorom sa vývojové prostredie nachádza po ukončení práce sprievodcu vytvorením reportu a zobrazením PREVIEW, v ktorom si môžeme prehliadnúť finálnu podobu report tak, ako bude v budúcnosti zobrazený u jeho „konzumentov“. Stredné okno vývojového prostredia sa v mode preview prepne do režimu prehliadača webového obsah



Vývojové prostredie v režime Preview

Ak sa nám podarí zobrazit preview reportu, znamená to, že všetky základné parametre a návrhové prvky sú nastavené a nakonfigurované správna (pripomíname, že zatiaľ nás zajíma len obsah, konečný dizajn reportu budeme ešte dotvárať). Môžeme teda report zostaviť a odoslať na reportovací server (menu Build, Deploy Solution), kde bude k dispozícii pre klientov. V okne Output môžeme sledovať protokol tejto akcie

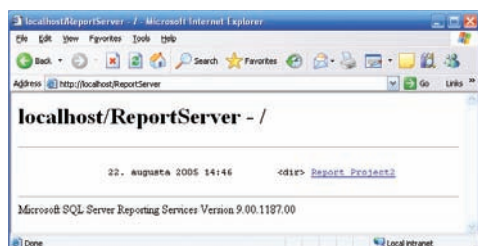
```
Build complete -- 0 errors, 0 warnings
----- Deploy started: Project: Report Project2, Configuration: Debug -----
Deploying to http://localhost/ReportServer
Deploying report ,/Report Project2/Report1\
Deploy complete -- 0 errors, 0 warnings
===== Build: 1 succeeded or up-to-date, 0 failed, 0 skipped =====
===== Deploy: 1 succeeded, 0 failed, 0 skipped =====
```

V okne Output je aj URL adresa, na ktorej je report prostredníctvom reportovacieho servera prístupný. Túto URL adresu môžeme zadať do prehliadača Internet Explorer a prezrieť si momentálnu podobu reportu, tak ako ju reportovací server poskytuje klientom.

## Rozhranie pre správu reportovacieho servera a reportov

Na tomto mieste po úspešnom prístupe k prvému reportu cez reportovací server si urobíme krátku odbočku ohľadne správy reportovacieho servera a reportov. Reportovacie služby si môžeme zjednodušene predstaviť ako bezstavový server, ktorý je súčasťou SQL Servera 2005 spravujúci metadáta (údaje o údajoch), definície objektov a podobne. Tieto údaje má vo vlastnej databáze, ktorá je taktiež pod správou SQL Servera 2005. Je implementovaný ako služba operačného systému Windows, rovnako ako napríklad webový server.

Report server je možné ovládať cez web – nachádza sa vo virtuálnom adresári a je prístupný cez URL adresu [http://nazov\\_pocitaca/ReportServer](http://nazov_pocitaca/ReportServer) ktorú môžeme prípadne modifikovať pri inštalácii. Prístup k lokálnemu serveru je možný aj cez adresu <http://localhost/ReportServer>



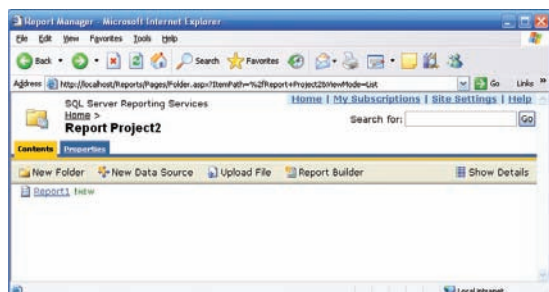
Test reportu pomocou webového prehliadača – zoznam reportov

Category	Sub Category	Sales
Accessories		152074.7600
	Bike Rack	51600.0000
	Bottles and Cages	1806.3800
	Cleaners	2838.1500
	Helmets	71379.6000
	Hydration	18586.6200

Test reportu pomocou webového prehliadača – zobrazenie reportu

## Report Manager

Nástroj Report Manager sa používa pre správu reportov. Nachádza sa vo virtuálnom adresári a je prístupný cez URL adresu <http://localhost/reports>, ktorú môžeme prípadne pri inštalácii zmeniť



Report Manager – prístup cez webový prehliadač

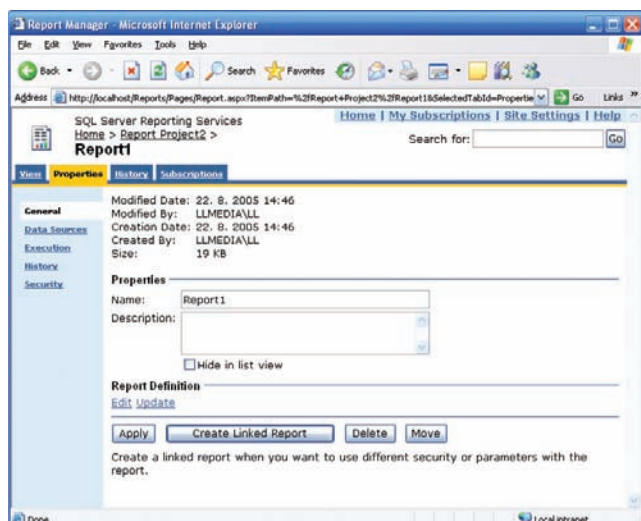
Na tejto stránke je prehľadne usporiadané niekoľko skupín ovládacích prvkov. Vpravo úplne hore je malý textový toolbar s položkami

- 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

Ak klikneme na názov reportu, zobrazí sa nám jeho preview, čiže uvidíme prakticky to isté ako v režime Preview vo Visual Studiu. Prechodom do záložky Properties dostávame na stránku správy reportu.



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

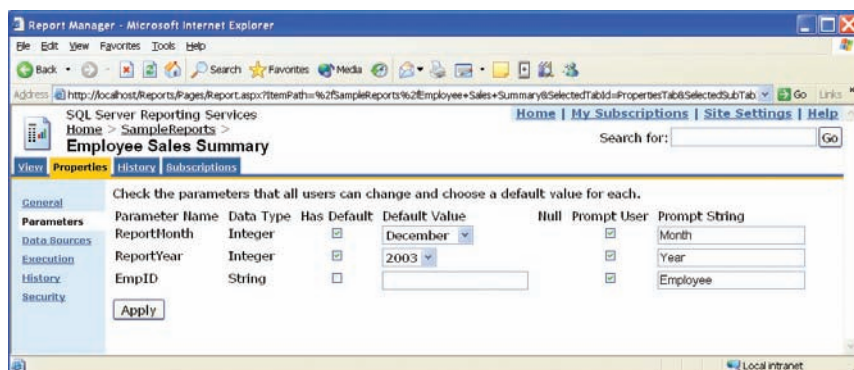
Jednotlivé záložky menu správy reportu si popíšeme podrobnejšie.

### General

Nosnou časťou tejto záložky je názov reportu a súhrnné informácie o ňom.

### Parameters

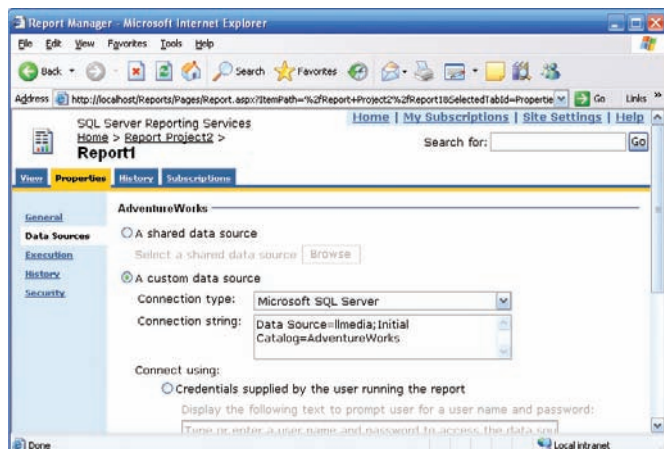
V záložke *Parameters* je nastavenie parametrov, pre reporty, ktoré tieto parametre vyžadujú. Niektoré parametre môžeme prednastaviť na nejaké implicitné – často používané hodnoty, napríklad časový interval na fiškálne obdobie, môžeme nastaviť aktuálny dátum a podobne. Na stránke sa zobrazí názov parametra 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 sa zobrazí na stránke reportu vedľa okna pre zadanie hodnoty.



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



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.

## History

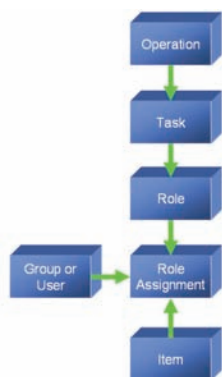
Ukladanie histórie reportu nie je principiálne zložité. Stačí ukladať časové snímky (snapshot) jednotlivých reportov, ktoré sú práve aktuálne. V záložke History definujeme spôsob generovanie snapshotov a počet snímok do minulosti, ktoré budú pre daný report uchované.

## Security

Záložka Security slúži pre zabezpečenie údajov v reporte. Bezpečnostný model reportovacích služieb je založený na roliach. 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



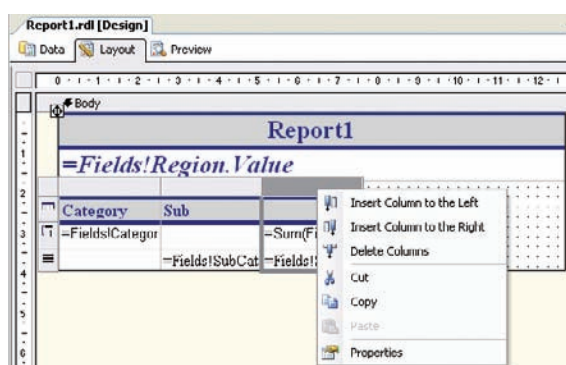
Bezpečnostný model založený na roliach

Pomocou tlačidla „Edit Item Security“ 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.

## Grafické úpravy reportu v prostredí Visual Studio

Report je pomerne zrozumiteľný, prehľadný a má dosť dobrú úpravu, no pravdepodobnosť, že úplne vyhoví našim požiadavkám a hlavne požiadavkám našich zadávateľov je pomerne malá. Sprievodca si na základe nami zadaných údajov urobil svoju robotu, no myšlienky zadávateľov jednoducho čítať nevie. Preto je potrebné v ďalšej návrhovej etape report požiadavkám prispôsobiť.

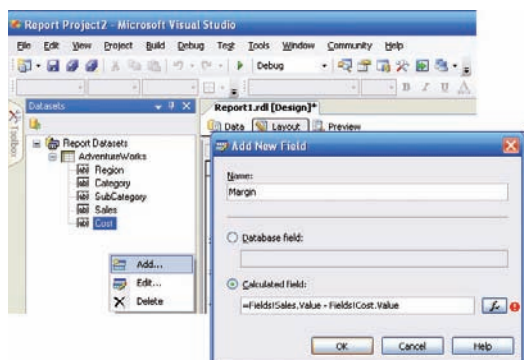
Ako prvú možnosť úprav ukážeme pridanie ďalšieho stĺpca do reportu. Postup je jednoduchý. Označíme stĺpec, s ktorým bude budúci stĺpec susediť a pomocou kontextového menu určíme, či novo vytvorený stĺpec bude situovaný vľavo, alebo vpravo od označeného stĺpca.



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.

V úpravách môžeme pokračovať napríklad pridaním vypočítaného poľa. V okne Datasets aktivujeme položku kontextového menu „Add“ a v dialógu „Add New field“ zadáme názov nového poľa a označíme prepínač „Calculated field“. Hodnotu poľa vypočítame ako rozdiel SALES – COST



Návrh nového stĺpca

V jazyku RDL to zapíšeme ako

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

Takýto 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

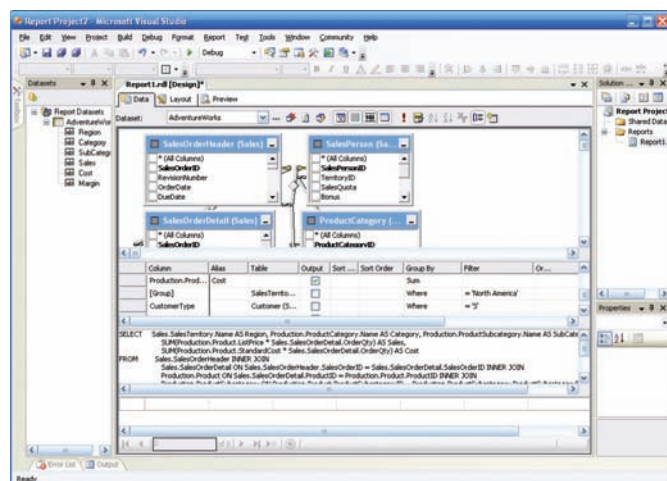
znamienko mínus, presunieme pole COST a návrh výrazu je hotový. Pomáha nám to hlavne pri dodržiavaní syntaxe. Report sprehľadníme aj vhodným naformátovaním výpisu hodnôt stĺpcov. V našom prípade je väčšina údajov v menových jednotkách, takže pre tieto stĺpce nastavíme vlastnosť formát na hodnotu C (z anglického currency). Pokračovať môžeme zvýraznením sumárnych polí a podobne.

## Interaktívny parametrický report

Reportovacie služby SQL Servera 2005 umožňujú vytvárať parametrické reporty. Takéto reporty môžeme nazvať aj interaktívne, pretože interaktívne umožňujú zobraziť len tú podmnožinu údajov, ktorú klient potrebuje. Ak chceme vytvoriť parametrický report, je potrebné principiálne zaviesť do SQL dopytu, ktorý je základom reportu parametre do podmienky.

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

Column	Filter
[Group]	= ,North America'
CustomerType	= ,S'



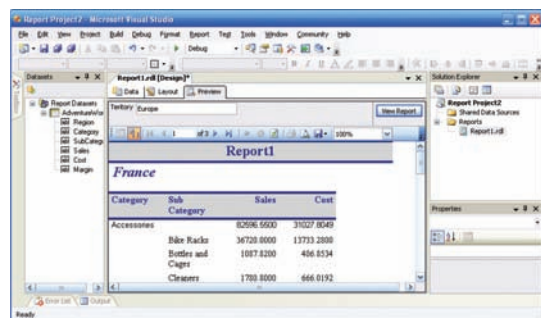
Pracovná obrazovka vývojového prostredia v režime „Data“

Všimnite si v na začiatku príkladu SQL dopyt, konkrétne podmienku za klauzulou WHERE.

```
...
WHERE SalesTerritory.[Group] = ,North America'
AND Customer.CustomerType = ,S'
```

Nahradíme v stĺpci Filter (riadok GROUP)reťazec = ,North America' reťazcom =@Teritory.

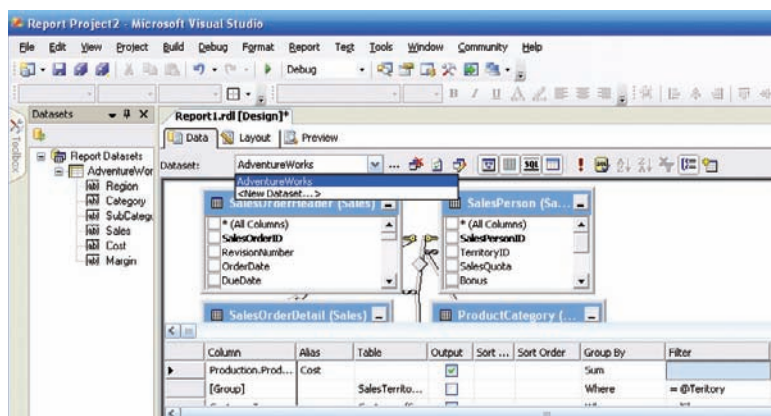
Skutočne postačí takýto jednoduchý návrhový úkon a máme k dispozícii parametrický report. V režime Preview sa zobrazí najskôr prázdny report, len s lištou pre zadanie parametra. Zadajme napríklad parameter Europe.



Preview parametrického reportu

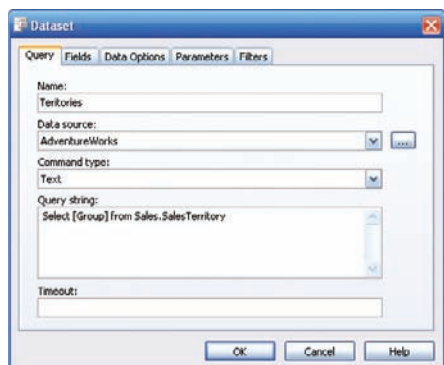


Veľmi jednoduché a efektné, no len v prípade ak vieme, čo zadať ako parameter. Oveľa lepšie by bolo zadávať teritória pomocou Comboboxu. Tento však treba naplniť údajmi zo stĺpca Group z tabuľky SalesTerritory. Všimnime si v záložke Data ľavé okno Datasets a combo box na lište. Report môže obsahovať viacero datasetov, pričom každý môže byť napojený na rôzne dátové zdroje. Nový dataset vytvoríme tak, že v hornej lište pracovnej obrazovky v móde Data aktivujeme možnosť „New Data Set“. Dataset nazveme napríklad „Territories“ a taktiež ho napojíme na databázu AdventureWorks.



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 „Territories“ a aktivujeme tlačidlo „Add Table“ (posledné tlačidlo toolbaru vpravo)



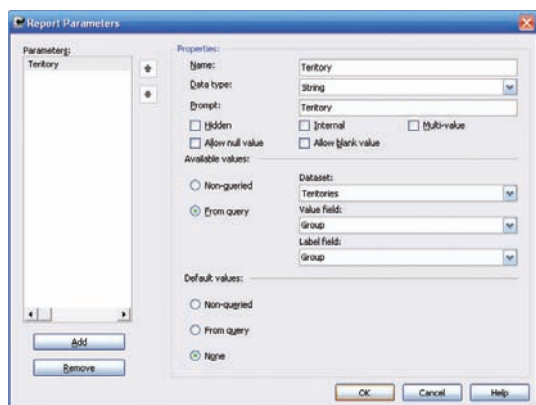
Parametre pre vytvorenie nového datasetu

Základom datasetu bude dopyt

```
Select distinct [Group] from Sales.SalesTerritory
```

Prepneme sa do módu Layouta aktivujeme v hlavnom menu položku „Report“ a v nej „Report Parameters“

Prepínač „Available values“ nastavíme na hodnotu „From Query“. Tým sa mierne zmení vzhľad dialógu. Pomocou comboboxu „dataset“ nastavíme tento parameter na dataset „Territories“ a „Value field“ prepneme na hodnotu „Group“. Parameter Default Values nastavíme na hodnotu „Non-queried“



Nastavenie parametrov

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

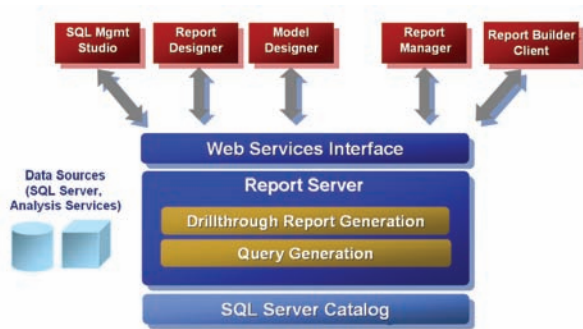
## Report Builder – nástroj pre vytváranie „ad – hoc reportov“

Ak sa pozrieme na obrázok rozdelenia ľudí využívajúcich reporty, zistíme, že popri najpočetnejšej skupine pasívnych konzumentov existujú aj používatelia, ktorí by si chceli pre nich generované reporty prispôbovať svojim momentálnym požiadavkám, prípadne predmetu skúmania. Do určitej miery im to umožňujú parametrické reporty, v tomto prípade však návrhová štruktúra reportu zostáva nezmenená, pomocou parametrov je možné upraviť len množinu údajov, ktorá bude do reportu zahrnutá. Reportovacie služby SQL Servera obsahujú aj nástroj Report Builder, pomocou ktorého je možné navrhovať reporty. No je tu jeden veľký rozdiel. Zatiaľ čo ak navrhujeme report v prostredí Visual Studio, vtedy kvalifikovaný vývojár potrebuje poznať štruktúru údajov. Poznáte to z predchádzajúcich príkladov, kedy bolo potrebné sformulovať SQL dopyt pre ich výber. Od bežného používateľa, napríklad analytika však nemôžeme očakávať, že bude poznať štruktúru údajov, nad ktorými bude report navrhovať. Pre Report Builder preto musia kvalifikovaní návrhári pripraviť akúsi modelovú medzivrstvu – „business model“ ktorý bude východiskom pre návrh reportov používateľom. Ten potom potrebuje poznať len štruktúru „business modelu“, ale nebude potrebovať poznať štruktúru údajov nad ktorými bol model vytvorený.



Filozofia a pozícia Report Buildera

Report Builder je klasická “ClickOnce” Windows aplikácia, ktorá je klientom reportovacích služieb. Na počítači na ktorom je spustená je potrebné mať nainštalovanú technologickú platformu .NET Framework 2.0. Pre návrh reportu je možné využiť bežné komponenty známe napríklad z kancelárskeho balíka Office, napríklad textboxy, tabuľky, matice, grafy a podobne. Po ukončení návrhu bude report umiestnený na server.

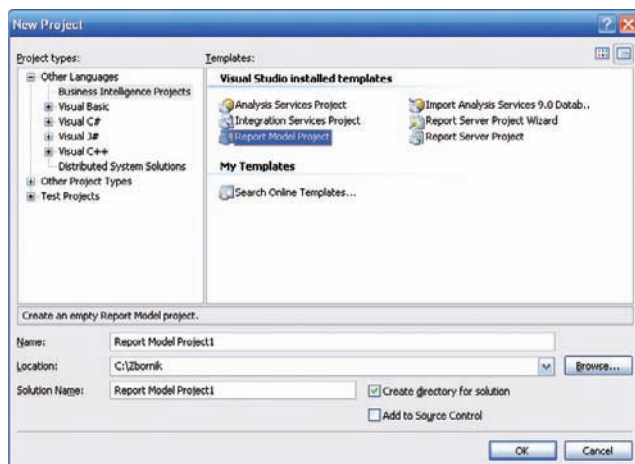


Report Builder – architektúra

Pomocou nástroja Report Builder môžeme vytvárať reporty, ktoré sú totožné s ostatným reportami pre Reportovacie služby. Používa sa jazyk RDL. Pre administráciu a zabezpečenie sa používajú rovnaké nástroje a aplikačné rozhrania ako pre klasické reporty. Na rozdiel od klasického reportu obsahuje report len metadáta, presnejšie sémantický model metadát.

## Vytvorenie modelu reportu

V zložke Business Intelligence Projects vytvoríme nový projekt podľa šablóny Report Model Project

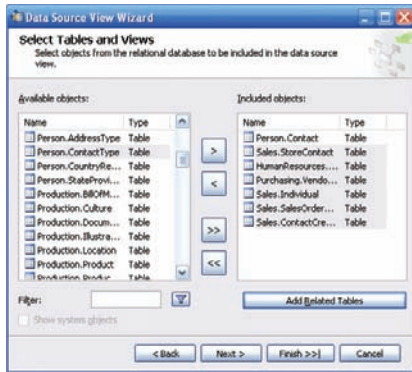


Vytvorenie nového projektu typu Report Model Project

Aj v tejto šablóne je postup vytvárania modelu daný zložkami v okne Solution Explorer

Data Sources  
Data Source Views  
Report Models

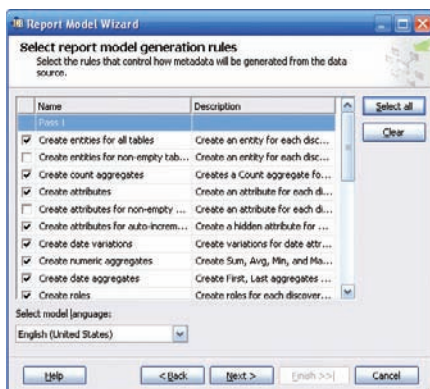
Ako zdroj údajov definujeme pomocou sprievodcu databázu AdventureWorks (pripojovací reťazec vygenerovaný sprievodcom bude `Data Source=llmedia;Initial Catalog=AdventureWorks;Integrated Security=True`).



Definovanie pohľadu na dáta – výber tabuľky *Person.Contact*. Pod ňou sú tabuľky relačne zviazané s touto tabuľkou

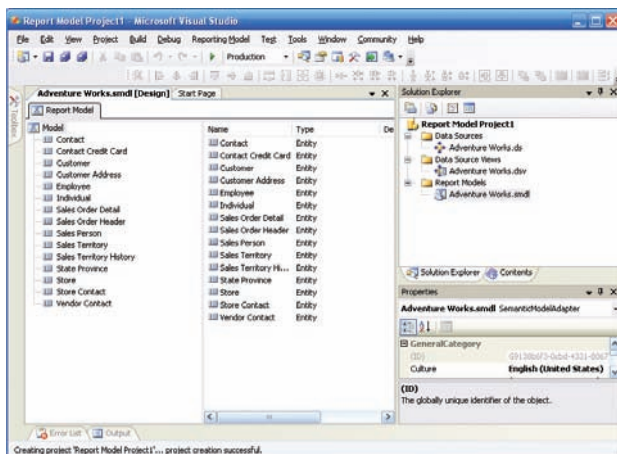
V sprievodcovi pre vytvorenie pohľadov na údaje vyberieme tabuľku *Person.Contact* a tabuľky s ňou relačne zviazané. Tie vyberieme pomocou tlačidla „Add Related Tables“. Podobne postupujeme aj pre tabuľku *Sales.Customer*. V zozname tabuliek relačne zviazaných s tabuľkou *Sales.Customer* je aj tabuľka *Sales.SalesTerritory*. V pravom okne ju označíme a vyberieme aj relačne zviazané tabuľky pre túto tabuľku. Nakoniec pridáme tabuľku *Sales.SalesOrderDetail* ale už bez tabuliek ktoré sú s ňou relačne zviazané.

Doteraz vykonávané činnosti sme popísali len heslovito, sú známe z predchádzajúcich projektov. Nasledujúca fáza – vytvorenie modelu je špecifická pre tento druh prípadov, kde sa predpokladá následné použitie nástroja Report Builder Pri vytváraní modelu preberá taktovku sprievodca Report Model Wizard. Po výbere pohľadu na zdroj údajov sa zobrazí dialóg Select report model generation rules. V ňom ponecháme implicitne označené položky



Dialóg „Select report model generation rules“ sprievodcu „Report Model Wizard“

V dialógu, ktorý bude nasledovať skontrolujeme, či je označená voľba „Update statistics before generating“.



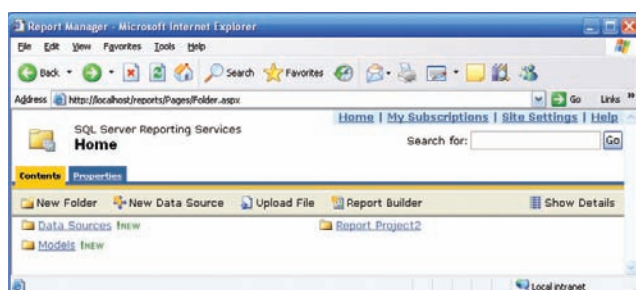
Model reportu zobrazený na pracovnej ploche vývojového prostredia

Projekt zostavíme a uložíme pod správu serveru. Obidva úkony spustíme pomocou zelenej šípky (Run) na toolbare. O podrobnostiach tohto aktu sa dozvieme z protokolu v okne Output

```
Build complete -- 0 errors, 0 warnings
--- Deploy started: Project: Report Model Project1, Configuration: Production ---
Deploying to http://localhost/ReportServer?%2f
Deploying data source ,/Data Sources/Adventure Works`.
Deploying model ,Adventure Works`.
Deploy complete -- 0 errors, 0 warnings
===== Build: 1 succeeded, 0 failed, 0 up-to-date, 0 skipped =====
===== Deploy: 1 succeeded, 0 failed, 0 skipped =====
```

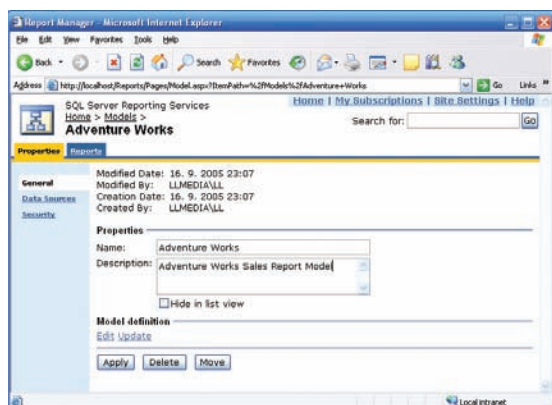
Ak sa deploy modelu skončil úspešne, to je vlastne finále našej činnosti vo Visual Studiu. Pre istotu si projekt uložíme.

Model máme vytvorený. O jeho zavedení pod správu reportovacích služieb sa môžeme presvedčiť v Internet Exploreri, kde navštívime URL adresu <http://localhost/reports/>. Už z úvodnej obrazovky je zrejmé, že pod správu reportovacieho servera pribudol nový model



Report Manager

Zobrazme si položku modelu a do políčka Description model nejakou charakterizujeme, aspoň stručne aby bolo zrejmé, čoho sa týka. V našom prípade by mohol byť dosť výstižný názov Adventure Works Sales Report Model

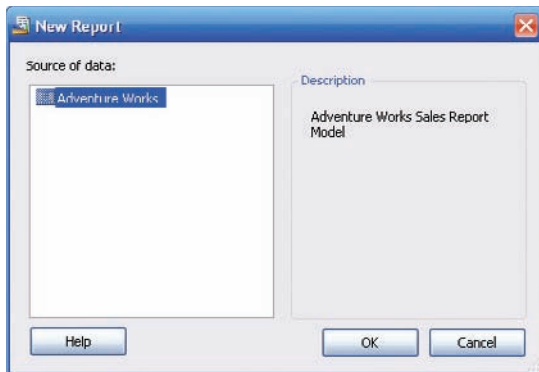


Stránka s vlastnosťami modelu. Doplnili sme jeho popis

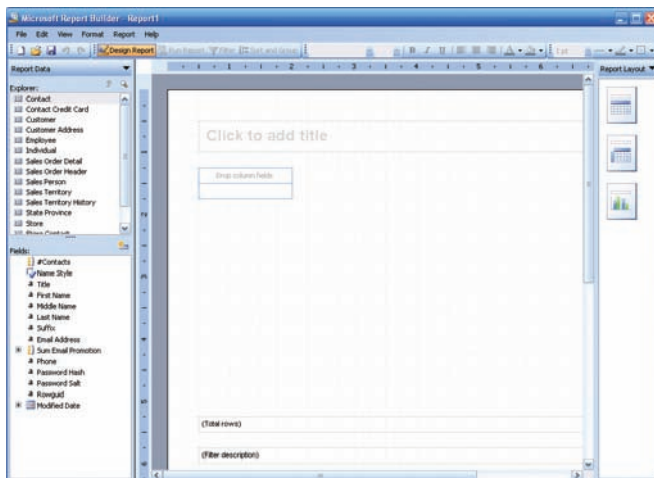
Pomocou odkazu Home sa môžeme vrátiť na hlavnú stránku modelu reportu, odkiaľ je možné stiahnuť spustiť Report Builder

## Návrh reportu pomocou Report Buildera

Po aktivovaní tlačidla Report Builder na hlavnej stránke reportu sa najskôr stiahne samotná aplikácia Report Builder zo servera na klientský počítač. Po spustení aplikácie sa zobrazí dialóg pre výber modelu. V prvom príklade máme situáciu jednoduchú, pod správou reportovacieho servera máme len jeden model, ktorý sme vytvorili podľa návodu v predchádzajúcej stati.



*Výber modelu ako zdroja údajov*

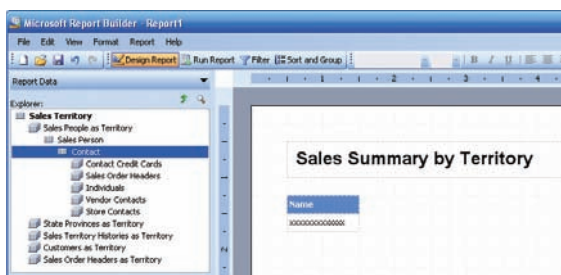


*Rozvrhnutie pracovnej plochy nástroja Report Builder*

Budovanie dizajnu reportu v prostredí aplikácie Report Builder začneme nahradením nápisu „Click to add title“ názvom reportu, napríklad „Sales Summary by Territory“

Entitu „Sales Territory“ z okna Explorer presunieme do poľa formulára označeného ako Drop column fields

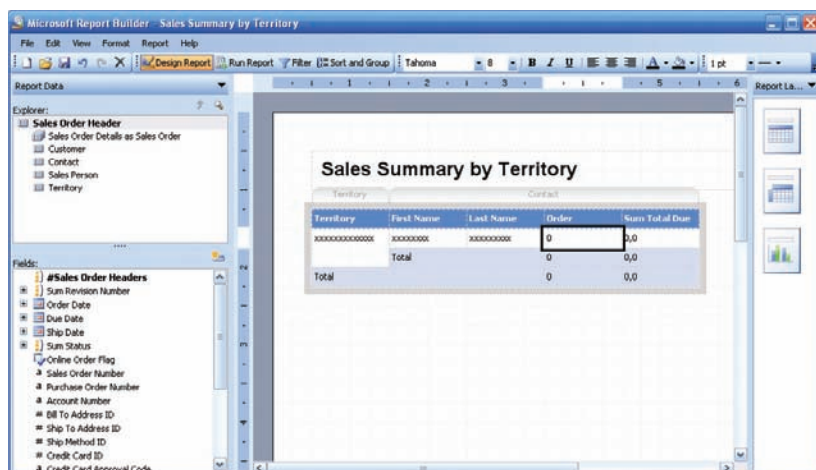
V okne Explorer sa zobrazí filter dostupných entít. V jeho stromovej štruktúre sa postupne navigujeme po položkách Sales People as Territory -- Sales Person --- Contact.



*Rozvrhnutie pracovnej plochy nástroja Report Builder*

V okne Fields postupne označíme položky First Name a Last Name a presunieme ich na plochu formulára vpravo od položky Territory.

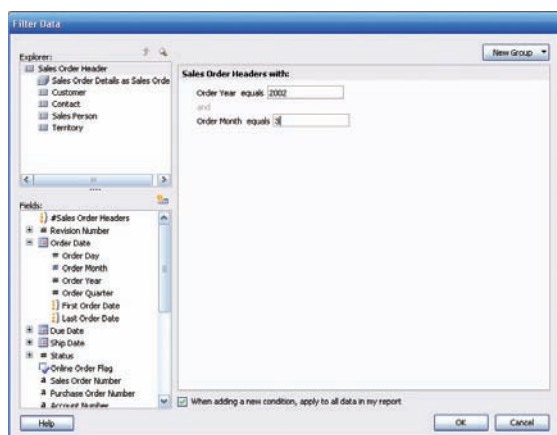
V okne Explorer označíme položku Sales Order Headers a z okna fields presunieme vpravo od doteraz navrhnutých stĺpcov položku #Sales Order Headers. Podobne pokračujeme aj s položkou Sum Total Due. Aby mal report prijateľnú podobu, premenujeme záhlavie stĺpca #Sales Order Headers na Orders. Do režimu editovania záhlavia stĺpca sa dostaneme kliknutím na políčko záhlavia.



*Navrhnutý report*

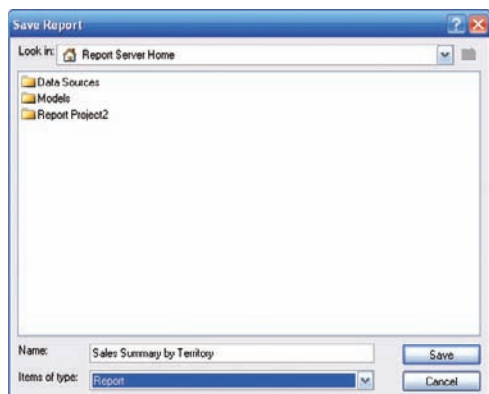
Rovnako premenujeme aj záhlavie Sum Total Due na Total Sales.

Všimnime si na toolbare tlačidlo Filter (ikona s lievikom). V ňom môžeme stanoviť podmienku pe obmedzenie množiny údajov, ktoré budú pomocou reportu zobrazené. V našom príklade sme chceli vybrať len údaje z marca 2002, preto sme nastavili v podmienke rok na hodnotu 2002 a mesiac na hodnotu 3.



*Dialóg filtra*

Podmienka sa formuluje presunutím príslušného atribútu na návrhovú plochu dialógu , nastavením porovnávacieho operátora a definovaním hodnoty, voči ktorej sa atribút porovnáva.



*Uloženie reportu na server*

Po uložení reportu na server ho môžeme spustiť buď priamo v prostredí Report Buildera, prípadne prezerať cez prehliadač webových stránok ako klasický report pod správou reportovacích služieb

**Sales Summary by Territory**

Sales People with: Any Sales Order Headers of Sales Person with All of (Order Year = 2002, Order #

Total Sales People: 10

Territory	First Name	Last Name	Orders	Total Due
	Stephen	Jiang	48	1513435.2384
	Total		48	1513435.2384
Canada	José	Saraiva	271	7967768.8006
	Garrett	Vargas	234	4840689.2515
	Total		505	12808458.0521
Central	Jillian	Carson	473	13434509.5476
	Total		473	13434509.5476
Northeast	Michael	Blythe	450	12433502.8355
	Total		450	12433502.8355
Northwest	David	Campbell	189	5029046.9145
	Pamela	Ansman-Wolfe	95	4453081.8838



## Kapitola 7: Klientský prístup k BI službám

Už názvov „Analytické služby“ a „Reportovacie služby“ vyplýva, že je použitá architektúra klient – server. Z toho vyplýva aj rozdelenie procesov pre spracovanie a prístup k údajom medzi servery a klientské počítače, na ktorých sa vykonáva prehliadanie výsledkov analýz, prípadne ich ďalšie spracovanie na lokálnej úrovni. Pre modelovanie a návrh multidimenzionálnych databáz sa využívajú grafické návrhové nástroje integrované s databázovými servermi, nakoľko túto činnosť vykonávajú hlavne databázoví administrátori analytici. Pre klientský prístup je potrebné uvažovať o inej filozofii. Zo skúseností totiž vieme, že dátové sklady a analytické databázy sú v podnikovom priestore efektívne len vtedy, ak ich môžu ľahko a jednoducho využívať príslušní klient, napríklad manažéri, analytici. Každý z nich potrebuje iný segment údajov, iný typ prístupu a inak prezentované údaje. Z pohľadu používateľa je najlepšia klientská aplikácia taká, ktorú tento dôverne pozná, teda ideálny prípad je integrácia prístupu k analytickým službám do aplikácií s ktorými klienti dosiaľ pracujú. Na jednej strane to môže byť prehliadač webového obsahu ako univerzálny tenký klient, opačným protipólom sú rôzne špeciálne jednoúčelové aplikácie.

### Excel ako klient BI služieb SQL Severu 2005

Ak sa zamyslíme nad tým, aký typ softvéru sa na klientských počítačoch najčastejšie používa, bezpochyby zvíťazia kancelárske balíky. Preto Microsoft v pomerne veľkom rozsahu integroval podporu analytických služieb do svojho kancelárskeho balíka Microsoft Office, hlavne do programu Excel.

Program Excel môže vystupovať voči OLAP serveru ako klient, prípadne je možné využiť jeho vlastný OLAP engine založený na službe Pivot Table Service. Pri klientskom prístupe je potrebná konektivita na analytický server. Ak aj uložíme pracovný hárok s výsledkom analýzy a neskôr ho otvoríme a budeme potrebovať napríklad meniť rozsah, alebo presnosť dimenzií, podarí sa nám to len za predpokladu, že budeme mať spojenie na OLAP server. Pri pripojení sa k OLAP serveru pomocou Excelu, vtedy zosumarizované hodnoty vypočítava OLAP server, Excel ako klientská aplikácia tieto údaje len zobrazuje, prípadne ďalej spracováva, alebo napríklad ukladá ako pohľad. Pri zobrazení, alebo zmene zostavy sa preto medzi OLAP serverom a klientskou aplikáciou posielajú pomerne málo údajov. Excel bol pre tento účel vybraný nielen kôli rozšírenosti, ale do hry vstupujú aj ďalšie dva nemenej podstatné skutočnosti: podpora kontingenčných tabuliek Office Web Components pre prístup z webu. Kontingenčná tabuľka (anglicky Pivot Table) má na rozdiel od klasickej tabuľky niekoľko špeciálnych vlastností. Napríklad umožňuje určitú rotáciu, teda výmenu riadkov a stĺpcov, kombináciu a hierarchickú štruktúru riadkov a stĺpcov. Vzhľadom k týmto vlastnostiam sa mimoriadne hodí pre zobrazenie údajov z multidimenzionálnych databáz. Do obdĺžnika pre polia údajov umiestnime fakty a do obdĺžnikov pre polia riadkov a stĺpcov umiestnime príslušné dimenzie. V zostave kontingenčnej tabuľky alebo kontingenčného grafu sa z každej dimenzie stáva množina polí, v ktorých možno rozbaľiť zbalíť podrobnosti na jednotlivých úrovniach hierarchií. Údajové polia sú polia zo zdrojového zoznamu, tabuľky alebo databázy, ktoré obsahujú údaje zosumarizované v zostave kontingenčnej tabuľky alebo kontingenčného grafu. Údajové polia zvyčajne obsahuje číselné údaje, napríklad štatistické údaje alebo čiastky predaja.

#### Režim On Line

V hlavnom menu programu Excel zvolíme položku menu *Data > Importovať externé údaje > Nový databázový dopyt*. Zobrazí dialóg pre pripojenie sa k externému zdroju údajov so záložkami Databázy, Dopyty a Datové kocky OLAP. Po pomenovaní zdroja údajov a výbere sprostredkovateľa, (implicitne Microsoft OLE DB Provider for OLAP Services) sa prepracujeme do sústavy dialógových okien pre zadanie údajov potrebných pre vybraného sprostredkovateľa. Môžeme sa pripojiť priamo na OLAP server, alebo môžeme načítať údaje OLAP kocky zo súboru. Po vytvorení zdroja údajov príde na radu ich zobrazenie, kedy je možné využiť služby Sprievodcu kontingenčnou tabuľkou.

Definovaním pripojenia a návrhom rozloženia hárku sa možnosti programu MS Excel zďaleka nekončia. Stačí napríklad kliknúť na ikonu grafu a máme výsledky analýzy v prehľadnej forme kontingenčného grafu, ktoré môžeme predložiť príslušnému manažérovi.

#### Práca s údajmi OLAP v režime off-line

OLAP kocky uložené v súboroch s príponou .CUB na pevnom disku počítača nám umožnia pracovať v režime off-line teda aj po odpojení od servera OLAP. Na základe týchto údajov môžeme vytvoriť zostavu kontingenčnej tabuľky alebo kontingenčného grafu.

## Jazyk MDX

Skratka MDX je skratka **M**ultidimensional **E**xpresions (multidimenzionálne výrazy). Tento jazyk je zjednodušene povedané určitým ekvivalentom jazyka SQL v relačných databázach. Jeho primárnym cieľom je navigácia v multidimenzionálnych údajoch. Pomocou MDX dotazu dokážeme vypísať vybranú podmnožinu údajov z multidimenzionálnej štruktúry (OLAP kocky) do dvojrozmernej tabuľky, ktorá obsahuje množinu buniek, preto tejto štruktúre hovoríme Cellset. Podobne ako jazyk SQL ani jazyk MDX nie je cieľom, ale len prostriedkom prístupu k multidimenzionálnym údajom. Buď pomocou klientskej konzoly, ale hlavne jazyk MDX rieši prístup k multidimenzionálnym údajom z rôznych aplikácií. Jazyk MDX môžeme používať v SQL Server Management Studiu, alebo priamo vo vyvíjaných aplikáciách cez rozhranie ADOMD.NET

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 [<specifikácia_osi>
      [, <specifikácia_osi>...]]
FROM [<specifikácie_kocky>]
[WHERE [<specifikácia_rezu>]]
```

Pravdepodobnejšie zrozumiteľnejší bude syntaktický zápis v mierne upravenom tvare

```
SELECT <<set>> ON COLUMNS,
      <<set>> ON ROWS,
      <<set>> ON PAGE
FROM <<cube>>
WHERE <<tuple>>
```

Mená objektov sa odporúča uzavrieť do hranatých zátvoriek [ ] napríklad:

```
SELECT { Measures.[Unit Sales] } ON COLUMNS
FROM sales
```

Niektoré názvy objektov, napríklad tie, ktoré sa skladajú z viacerých slov oddelených medzerou, napríklad [Unit Sales] alebo názvy objektov, ktoré sú totožné s kľúčovými slovami jazyka MDX napríklad [SELECT] alebo čísla napríklad [2003] v hranatých zátvorkách byť musia. Taktiež je potrebné používať plne kvalifikované mená objektov, napríklad prvý kvartál [Q1] môže byť v roku 2000, ale aj v roku 2002, 2003... Preto je potrebné používať úplné mená objektov napríklad [2002].[Q1] Ak plne kvalifikované mená nepoužijeme bude pri interpretácii príkazu použitý prvý výskyt daného mena objektu v kocke.

Podrobný popis jazyka MDX je mimo rozsah tejto publikácie, v krátkom prehľade uvedieme aspoň stručné definície základných pojmov

**Prvok (Member):** Definícia tohoto pojmu je jednoduchá a stručná. Môže ísť o ktorýkoľvek prvok (objekt) v hierarchii, napríklad

```
[Drink]
[2001]
[2001].[Q1].[Jan]
```

**N-tica (tuple):** je jeden prvok, prípadne prienik dvoch alebo aj viacerých prvkov. Napríklad

```
([Product].[Drink].[Alcoholic Beverages], [Customers].[USA].[CA])
[Product].[Food]
```

**Množina (set):** Týmto pojmom označujeme množinu n-tíc alebo prvkov Prakticky je to vlastne množina buniek vo výslednej tabuľke)

```
SELECT {[William Wong], [Andrea Donlon]} ON COLUMNS
FROM sales
```

```
SELECT {[Non-Consumable],USA }, (Beverages, Mexico) ON COLUMNS
FROM sales
```

Množiny nám vracajú aj niektoré funkcie. Platí pravidlo, že každý prvok množiny musí mať rovnaký počet dimenzií

### Dimenzie (osi tabuľky buniek)

Dimenzia je množina elementov rovnakého typu. Predpokladáme, že po preštudovaní kapitoly OLAP základnú terminológiu máme zvládnuť. Ako príklad pre výpis „dvojrozmernej“ tabuľky údajov ukážeme príkaz

```
SELECT
  {[Product].[All Products].[Drink], [Product].[All Products].[Food],
  [Product].[All Products].[Non-consumable] } ON COLUMNS,
  {[Customers].[Country].[USA].[CA], [Customers].[Country].[USA].[OR],
  [Customers].[Country].[USA].[WA]} ON ROWS
FROM Sales
WHERE ([Gender].[F] )
```

### MDX príkaz SELECT

Povinnou časťou MDX príkazu SELECT je výber príslušnej kocky pomocou klauzule FROM. V našom prípade vyberáme údaje z kocky SALES.

```
FROM Sales
```

MDX príkaz SELECT taktiež umožňuje definovať podmienku pre reštrikciu alebo v prípade kocky by bol vhodnejší termín výrez (obmedzenie množiny údajov). Ak chceme vybrať ženy, použijeme podmienku

```
WHERE ([Gender].[F] )
```

Pomocou klauzúl ON COLUMNS a ON ROWS sme definovali 2 dimenzie. Jazyk MDX umožňuje prácu až so 128 dimenziami. Tieto môžeme definovať buď slovné (COLUMNS, ROWS, PAGES, SECTIONS, CHAPTERS) alebo číselne ako AXIS (<index>).

Syntaktický predpis pre definovanie osí je:

```
<specifikace_osi> ::= <set> ON <jmeno_osi>
```

```
<jmeno_osi> ::= COLUMNS | ROWS | PAGES | SECTIONS | CHAPTERS | AXIS(<index>)
```

Príkaz

```
SELECT
  {[Product].[All Products].[Drink], [Product].[All Products].[Food],
  [Product].[All Products].[Non-consumable] } ON COLUMNS,
  {[Customers].[Country].[USA].[CA], [Customers].[Country].[USA].[OR],
  [Customers].[Country].[USA].[WA]} ON ROWS
FROM Sales
```

by sme teda mohli zapísať pomocou čísel osí aj nasledovne,

```
SELECT
  {[Product].[All Products].[Drink], [Product].[All Products].[Food],
  [Product].[All Products].[Non-consumable] } ON AXIS(0),
  {[Customers].[Country].[USA].[CA], [Customers].[Country].[USA].[OR],
  [Customers].[Country].[USA].[WA]} ON AXIS(1)
FROM Sales
```

**Rezy – klauzula WHERE**

Na rozdiel od jazyka SQL, kde sa klauzula WHERE používa pre definíciu reštrikcie, to jest pre obmedzenie množiny údajov, v jazyku MDX sa klauzula WHERE používa pre definíciu výrazu údajov. Ako uvidíme ďalej, pre obmedzenie množiny údajov sa v jazyku MDX používa operátor Filter().

Použitie klauzuly WHERE ukážeme na praktickom príklade.

```
SELECT {[Measures].[Unit Sales]} ON COLUMNS,
        {[Product].[Product Family].Members} ON ROWS
FROM Sales
```

**Redukcia údajov – operátor Filter()**

V závislosti od požiadaviek na zobrazené údaje je často potrebné obmedziť množinu údajov. Pre tento účel sa používa operátor Filter(), pričom jeho syntax je:

```
Filter (<<set>>, bool podmienka)
```

Pri konštrukcii podmienky môžeme používať tieto porovnávacie operátory

Porovnávací operátor	Význam operátora
=	rovnosť
<>	nerovnosť
>	väčší
<	menší
>=	väčší alebo rovný
<=	menší alebo rovný

a nasledovné logické operátory pre zostavenie zložitejšej podmienky na základe kombinácie viacerých jednoduchých podmienok

- NOT** Kombinácia sa uplatní ak nasledujúca podmienka je nepravdivá
- AND** Kombinácia sa uplatní ak obidve podmienky sú pravdivé
- OR** Kombinácia sa uplatní ak aspoň jedna podmienka je pravdivá

Ukážeme to na jednoduchom príklade, kde vypíšeme všetky (neprázdne) hodnoty Unit Sales pre jednotlivé obchody

```
SELECT {[Measures].[Unit Sales]} ON COLUMNS,
        NON EMPTY {[Store].[Store Name].members} ON ROWS
FROM [Sales]
```

**Rozhranie ADOMD.NET**

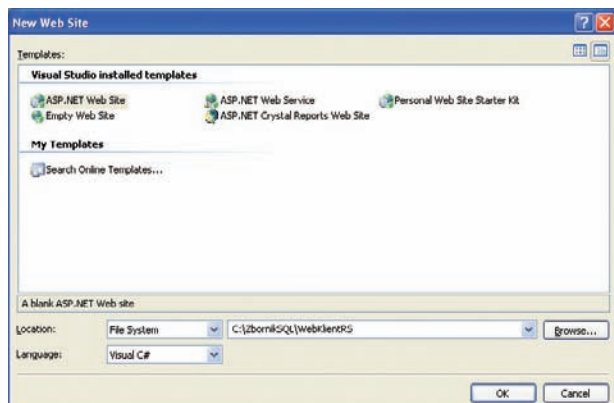
Pre vytvorenie klientskej aplikácie, ktorá bude pristupovať k údajom v OLAP kocke potrebujeme poznať aj rozhranie pre prístup k týmto údajom. Na platforme SQL Servera 2005 plní túto úlohu rozhranie ADOMD.NET (**A**ctiveX **D**ata **O**bjects (**M**ultidimensional)). Jedná sa o rozšírenie rozhrania ADO o podporu práce s multidimenzionálnymi údajmi.

**Klientské aplikácie k reportovacím službám vo Visual Studiu 2005**

Základné rozdelenie klientských aplikácií je podľa kritéria, či jadro aplikácie beží na serveri, alebo na klientskom počítači rozdeľujeme tieto aplikácie na takzvaného „tenkého“ a „bohatého“ klienta. V prípade obidvoch typov aplikácií nám zobrazenie reportu pod správou reportovacích služieb, prípadne lokálneho reportu značne uľahčí vizuálna komponenta Report Viewer. Tento prvok okrem práce s reportom pod správou reportovacích služieb dokáže pracovať aj v lokálnom režime so súbormi reportov (súbor s príponou RDLC)

## Webová aplikácia – tenký klient

Pod pojmom „**tenký klient**“ rozumieme spravidla počítač, kde sa pre prístup k údajom používa prehliadač webových stránok.

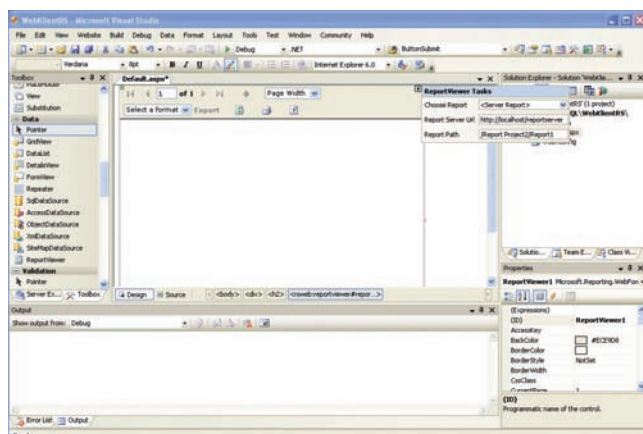


Vytvorenie projektu ASP.NET Web site

V procese vizuálneho návrhu formulára určíme kde bude umiestnený priestor pre zobrazenie reportu a na toto miesto aplikačného formulára umiestnime prvok ReportViewer. Všimnime si malú šípku v jeho pravej hornej časti. Pomocou nej sa aktivuje tasks menu. Pomocou neho je potrebné nastaviť parametre pre prístup k reportovaciemu serveru. V našom prípade sme nastavili URL adresu a cestu k reportu nasledovne:

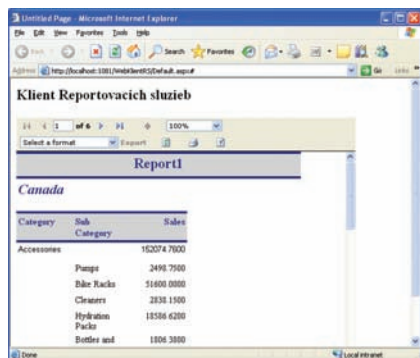
URL: <http://localhost/reports>

Path: /Report Project2/Report1



Pridanie prvku Report Viewer na plochu formulára webovej aplikácie

Po spustení aplikácie sa zobrazí report v okne prehliadača webových stránok.

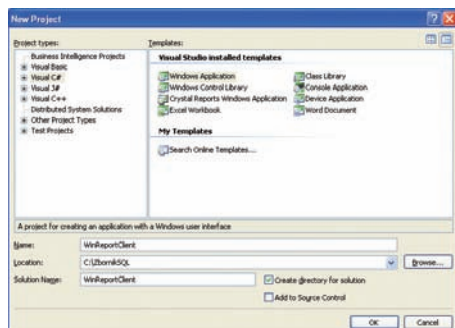


Spustenie webovej ASP.NET aplikácie využívajúcej reportovacie služby

## Windows aplikácia – bohatý klient (rich client)

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 služby rôznych serverov.

Vďaka komponente Report Viewer, ktorú nájdeme v Toolboxe, v podzložke Data je vytváranie klientských Windows aplikácií pre reportovacie služby veľmi jednoduché.

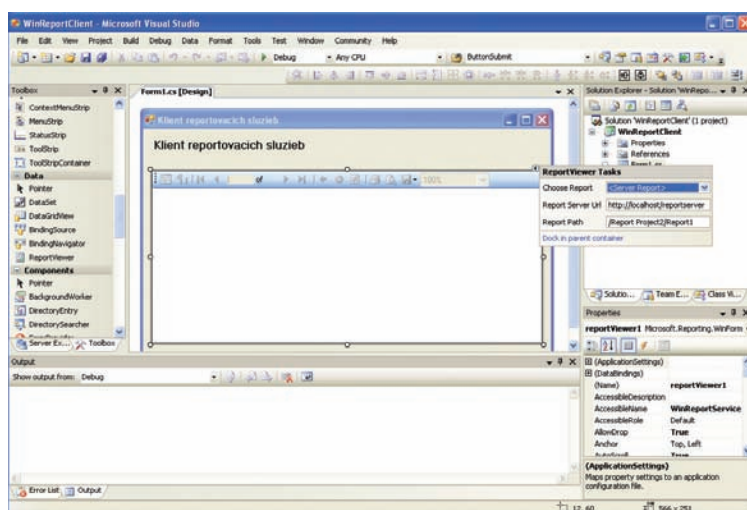


Vytvorenie projektu Windows Application

Celý proces vytvorenia prvku pre zobrazenie reportu sa zaobíde bez programovania. Stačí na plochu aplikačného formulára umiestniť spomínaný prvok a v jeho tasks menu nastaviť parametre. V našom prípade sme chceli zobraziť report z príkladu v kapitole o Reportovacích službách, ktorý je pod správou reportovacieho serveru. Po nastavení parametrov, v našom prípade

URL: <http://localhost/reports>

Path: /Report Project2/Report1



Pridanie prvku Report Viewer na plochu formulára aplikácie

môžeme aplikáciu preložiť a spustiť. Ak sú splnené všetky nevyhnutné podmienky (reportovací server je spustený a report ne zadanej adrese existuje a je správny), zobrazí sa report na ploche vizuálnej komponenty Report Viewer na ploche aplikačného formulára. Celý postup vytvorenia Windows aplikácie je dostatočne jasný s formulára.

Klient reportovacích služieb

Klient reportovacích služieb

Report1

Canada

Category	Sub Category	Sales
Accessories		152074.7600
	Pumps	2498.7500
	Bike Racks	51600.0000
	Cleaners	2838.1500

*Spustenie klientskej Windows aplikácie využívajúcej reportovacie služby*

## Zoznam použitej a doporučenej literatúry

- [1] Lacko, L.: Databáze: datové sklady, OLAP a dolování dat, Computerpress Brno 2004
- [2] Matiaško, K; Vnuk, L; Ševčíková, K: Dátové sklady ako informačný zdroj pre podporu rozhodovania, Žilinská univerzita
- [3] Rud, Olivia Parr: Data mining, Computerpress 2001
- [4] Humphries, M.: Data warehousing, Návrh a implementace, Computerpress 2001



Vydal:

Microsoft s.r.o., BB Centrum, budova Alpha, Vyskočilova 1461/2a, 140 00 Praha 4  
tel.: +420-261 197 111 , fax: +420-261 197 100, <http://www.microsoft.com/cze>