Mendelova univerzita v Brně
Provozně ekonomická fakulta

Využití potenciálu BI sémantického modelu v MS SQL Server 2012

Diplomová práce

Vedoucí práce:
Ing. Jan Přichystal Ph.D.

Bc. Jindřich Zelený

Brno, 2015

Rád bych na tomto místě poděkoval vedoucímu mé práce panu Ing. Janu Přichystalovi, Ph.D. za odborné a svědomité vedení, cenné rady a věcné připomínky. Dále bych rád poděkoval své rodině a přítelkyni za podporu při mých studiích.

Čestné prohlášení

Prohlašuji, že jsem tuto práci: **Využití potenciálu BI sémantického modelu v MS SQL Server 2012**
vypracoval samostatně a veškeré použité prameny a informace jsou uvedeny v seznamu použité literatury. Souhlasím, aby moje práce byla zveřejněna v souladu s § 47b zákona č. 111/1998 Sb., o vysokých školách ve znění pozdějších předpisů, a v souladu s platnou *Směrnicí o zveřejňování vysokoškolských závěrečných prací.*
Jsem si vědom, že se na moji práci vztahuje zákon č. 121/2000 Sb., autorský zákon, a že Mendelova univerzita v Brně má právo na uzavření licenční smlouvy a užití této práce jako školního díla podle § 60 odst. 1 Autorského zákona.
Dále se zavazuji, že před sepsáním licenční smlouvy o využití díla jinou osobou (subjektem) si vyžádám písemné stanovisko univerzity o tom, že předmětná licenční smlouva není v rozporu s oprávněnými zájmy univerzity, a zavazuji se uhradit případný příspěvek na úhradu nákladů spojených se vznikem díla, a to až do jejich skutečné výše.

V Brně dne 1. ledna 2015 .....

Abstract

Zelený, J., Using the potential of BI semantic model in MS SQL Server 2012

This thesis deals with different approaches in the development of an analytical model of the data warehouse with a focus on tabular mode and its associated technologies and tools from the Microsoft company. In the theoretical part the thesis introduces the principles of Business Intelligence and also the concept of the semantic model. It also states tabular model as a new approach of creating an analytical database stored in the RAM memory. The tabular model is developed on the top of the data warehouse of a fictitious company Contoso in the practical part. The emphasis is put mainly on the comparison between the tabular and multidimensional model. The work ends with deploying both models on a virtual server with a comparison of their computing power for each of the designed scenarios.

Abstrakt

Zelený, J., Využití potenciálu BI sémantického modelu v MS SQL Server 2012. Diplomová práce. Brno, 2015

Diplomová práce se zabývá rozdílnými přístupy při tvorbě analytického modelu nad datovým skladem se zaměřením na tabulární režim a jeho přidružené technologie a nástroje od firmy Microsoft. Práce v teoretické části seznamuje čtenáře s principy Business Intelligence a představuje pojem sémantický model. Rovněž uvádí tabulární model, jakožto nový přístup k tvorbě analytické databáze uložené v RAM paměti. V praktické části je pak tento tabulární model vytvořen nad datovým skladem fiktivní firmy Contoso. Důraz je kladem především na srovnání tabulárního modelu s multidimenzionálním. Práce je zakončena nasazením obou modelů na virtuální server, kde je porovnáván jejich výpočetní výkon u jednotlivých navržených scénářů.

Obsah

1 Úvod a cíl práce

1.1 Úvod do problematiky

Oblast Business Intelligence (dále také BI), která poskytuje podporu pro rozhodování (především strategické), není na poli informatiky žádným nováčkem. Tato problematika je v kontextu informačních systémů (DSS[1]) skloňována již od 50. let 20. století. Termín Business Intelligence byl potom v roce 1989 blíže definován analytikem společnosti Gartner group Howardem Dresnerem jako „sada konceptů a metod určených pro zkvalitnění rozhodnutí firmy“. Konkurenční výhoda prostřednictvím snižování informační entropie stále zůstává pro podniky jedním z vysoko postavených cílů a BI systémy se tak staly nepostradatelným prvkem pro většinu z nich. Byly to však poslední dvě desítky let, ve kterých bylo možno zaznamenat v tomto dynamickém odvětví velmi progresivní růst, k němuž došlo především díky rozšíření datových skladů, tržišť a objemu do nich ukládaných dat. V současné době se nicméně růst BI odvětví dle společnosti Gartner (zabývající se IT výzkumem a poradenstvím) spíše zpomaluje, stabilizuje a do popředí se dostávají požadavky na inovativní prvky BI.

Množství dat, které podniky uchovávají, roste skokově, a proto nabývá požadavek na rychlé zpracování velkého objemu dat na důležitosti. Také propojení BI s moderními technologiemi, jako jsou smartphony, či sociální sítě, vyžaduje určitou flexibilitu použití, proto je vhodné při vytváření BI řešení uvažovat také s alternativními přístupy. Jedna z možností, jakou lze v některých případech dosáhnout velmi dobrých výsledků, je využití tzv. „in-memory“ technologií. Tento koncept umožňuje eliminovat snížení výpočetní rychlosti klasického přístupu způsobeného uložením dat na disku, neboť je možné uložit poměrně rozsáhlou databázi (rámcově stovky GB až několik TB) do operační paměti a pracovat s ní přímo, tedy bez nutnosti přístupu na toto datové úložiště. Volba struktury dat a způsob, jak s nimi bude nakládáno, je obvykle jedna z nejdůležitějších částí projektu.

Tato práce se zabývá právě konceptem „in-memory“ ve spojení s produkty firmy Microsoft, která tento koncept nazývá „Tabular model“ (podle využití sloupcové databáze) a plně podporuje v SQL Serveru od roku 2012. Datovou doménou je vzorová databáze Contoso Retail rovněž od firmy Microsoft. Tato databáze obsahuje data shromážděná fiktivní maloobchodní firmou a jsou navržena speciálně pro BI účely tak, aby simulovala komplexnost reálných dat a bylo na nich možno prezentovat sílu BI nástrojů.

[1] Systémy na podporu rozhodování (z angl. Decision Support Systems) pomáhají svým uživatelům (většinou manažerům) při realizaci řídících a rozhodovacích činností v podnikání. Uživatel tu může srovnávat dílčí výsledky řešení se svými představami a podle toho ovlivňovat další průběh řešení

1.2 Cíl práce

Cílem této práce je demonstrovat možnosti využití tabulárního režimu při tvorbě datových skladů na základě teoretických konceptů stojících za „in-memory“ technologiemi. Primárním cílem práce je vytvoření reálně využitelného BI řešení postaveného na tabulárním modelu. Sekundárním cílem je pak srovnání a analýza řešení postaveného na tabulárním a multidimenzionálním modelu, a to z hlediska výpočetní efektivity, vhodnosti nasazení a časové náročnosti v průběhu celého vývojového cyklu projektu.

2 Metodika řešení

Metodika řešení diplomové práce je rozdělena do dvou logických celků.

V teoretické části práce je představen pojem Business Intelligence jako vědní disciplína včetně jejích komponent a architektur, vysvětlen sémantický model BI a definovány trendy s ním související, provedena studie v současnosti využívaných BI nástrojů firmy Microsoft a v neposlední řadě obsahuje také srovnání těchto modelů, respektive architektur.

V praktické části práce jsou pak uplatněny tyto teoretické koncepty a demonstrovány na ukázkovém BI řešení ve formě obrázků, tabulek, grafů, či kompletních dashboardů[2]. Dále jsou nasazena tabulární a multidimenzionální BI řešení na virtuální server a otestována z hlediska výpočetní výkonnosti.

Práce končí celkovým shrnutím dosažených výsledků při tvorbě BI projektu a srovnáním efektivity využití multidimenzionálního a tabulárního přístupu k jeho řešení.

Jednotlivé kroky metodiky

- definice fundamentálních konceptů a pojmů z oblasti Business Intelligence,
- vysvětlení sémantického modelu,
- popis tabulárního a multidimenzionálního modelování,
- seznámení s nástroji BI vytvořenými firmou Microsoft a zkoumání podstaty evoluce jejich analytických služeb,
- tvorba a porovnávání tabulárního a multidimenzionálního modelu,
- nasazení modelů na virtuální server a porovnání jejich výpočetní výkonnosti

[2]Vizuální zobrazení nejdůležitějších informací, které jsou potřebné pro dosažení jednoho, nebo více cílů na jediné obrazovce monitoru tak, aby bylo umožněno snadné sledování.(Cancel, 2010)

3 Principy Business Intelligence

Tato kapitola definuje fundamentální koncepty a pojmy z oblasti Business Intelligence. Představuje hlavní komponenty BI, jako je datový sklad, datové tržiště, dočasné úložiště dat, nástroje datové integrace, a další.

3.1 Definice Business Intelligence a komponent BI

3.1.1 Business Intelligence

Pojem Business Intelligence je v české literatuře nejčastěji definován jako sada procesů, aplikací a technologií, jejichž cílem je účinně a účelně podporovat rozhodovací procesy ve firmě. Podporují analytické a plánovací činnosti podniků a organizací a jsou postaveny na principech multidimenzionálních pohledů na podniková data (Novotný, Pour, Slánský, 2005). BI systémy slouží, na rozdíl od běžných transakčních systémů, primárně k podpoře rozhodovací činnosti. BI systémy poskytují ve vhodné formě agregovaná analytická data odvozená z dat podnikových informačních systémů a pomáhají jejich uživatelům proniknout do podstaty složitých ekonomických jevů, pro jejichž analýzu bylo BI nasazeno. BI systémy umožňují odhalit vztahy v ekonomické realitě a sledovat vývoj významných podnikových indikátorů (ukazatelů) z pohledu jejich relevantních dimenzí, např. indikátor vývoje podnikových tržeb z pohledu členění v jednotlivých regionech. Díky analytickému zaměření umožňují BI systémy podnikům výrazně zvýšit efektivnost svých procesů a pomoci jim při rozhodování o strategických otázkách (Závodný, 2012). Oblast Business Intelligence sestává ze čtyřech základních komponent: architektura IS, modelování, controlling a plánování. Mezi nástroje BI patří zejména manažerské informační systémy (MIS), informační systémy pro podporu řízení (EIS), analytické kostky (OLAP), DSS, datamining (dolování dat), data warehousing (budování datových skladů) a reporting.

3.1.2 Datový sklad

Datový sklad (také datawarehouse, či DWH) je integrovaný, subjektově orientovaný, stálý a časově rozlišený souhrn dat, uspořádaný pro podporu potřeb managementu (Inmon, 2005). Pro lepší pochopení této definice lze blíže popsat podstatné vlastnosti datového skladu následovně (Novotný, Pour, Slánský, 2005).

- Integrovaný – data jsou ukládána v rámci celého podniku, a ne pouze v rámci jednotlivých oddělení.
- Subjektově orientovaný – data jsou rozdělována podle jejich typu, ne podle aplikací, ve kterých vznikla. Jde tedy o případ, kdy jsou data o zaměstnanci uložena pouze jednou, a to v jedné databázi datového skladu, kdežto v produkčním systému bývají rozptýlena do různých souborů podle toho, pro kterou aplikaci mají být použita.

- Stálý – datové sklady jsou koncipovány jako „Read-Only“, což znamená, že zde žádná data nevznikají ručním pořízením, a nelze je ani žádnými uživatelskými nástroji měnit. Data jsou do DWH načítána z operativních databází či jiných externích zdrojů a existují zde po celou dobu života datového skladu.
- Časově rozlišený – aby bylo možné provádět analýzy za určitá období, je nutné, aby byla do DWH uložena i historie dat. Načítaná data s sebou tedy musí nést i informaci o dimenzi času.

3.1.3 Datové tržiště

Na rozdíl od DWH jsou datová tržiště (datamarty) zaměřena pouze na jednu konkrétní cílovou skupinu uživatelů. Tyto cílové skupiny přitom vyžadují informace o konkrétní oblasti, konkrétním předmětu zájmu – tržiště tedy ve svých databázích obsahují přesně zacílenou podmnožinu dat vyhovující konkrétním analytickým požadavkům skupiny uživatelů. Data uložená v datových tržištích je možné používat především jako podklad pro cílené analýzy. Na základě těchto cílených analýz je pak pro odpovědné pracovníky mnohem jednodušší přijímat informovaná rozhodnutí, v návaznosti na zaměření tržiště jak taktického, tak strategického charakteru. Přínos datových tržišť je ovšem nejen v tom, že umožňují pomocí odpovídajících nástrojů získávat dříve neviděné informace, ale také v tom, že informace získatelné z běžných provozních systémů jsou s jejich pomocí dostupné jednodušším a rychlejším způsobem. Řízení podniku s dostatečným množstvím správných informací by mělo být efektivnější. (Scheps, 2013)

3.1.4 Nástroje datové integrace

- ETL – zkratka je tvořena počátečními písmeny třech slov, a to extrakce, transformace, load. Extrakce představuje získání dat z primárních systémů. V typickém podniku je takovýchto systémů, zajišťujících životně důležité operace, celá řada. Zpravidla byly nasazovány v různých časových etapách a využívají různé, někdy i vysoce proprietární a málo otevřené technologie. Získat data z těchto systémů někdy může být nesnadný úkol, zvláště když si uvědomíme, že se nejedná o jednorázovou akci, ale o periodickou činnost, jejíž cílem je poskytovat aktuální data pro další zpracování v systémech datových skladů. Transformace představuje proces zpracování dat získaných z primárních systémů do formy, odpovídající požadavkům systémů datových skladů. Transformace zahrnuje celou škálu operací od konverzí, matematických operací, filtrování, normalizace a denormalizace, až po sofistikované metody vytváření multidimenzionálních struktur. Data přicházející z primárních systémů mohou být (a ve valné většině případů jsou) znečištěna různými typy chybných či nekompletních údajů. Součástí transformací proto bývá také mechanizmus kontroly kvality dat a čištění dat. Výstupem transformačního procesu jsou korektní a konsolidovaná data, která mají maximální informační hodnotu. Závěrečnou fází je load, neboli na-

plnění zpracovaných dat do cílového systému datového skladu. Zde jsou pro uložení dat často využívány specializované technologie, které předpokládají použití různých proprietárních mechanizmů pro rychlé a optimální vkládání dat (Schiller, 2003).

- EAI – cílem nástrojů EAI je integrace primárních podnikových systémů a razantní redukce počtu jejich vzájemných rozhraní. Hlavním rozdílem mezi ETL a EAI nástroji je skutečnost, že EAI platformy pracují v reálném čase. EAI tak doplňuje dávkový přenos a umožňuje vznik nové generace datových skladů, tzv. Real-Time Data Warehouse (Novotný, Pour, Slánský, 2005).

3.1.5 Dočasné úložiště dat – DSA

DSA slouží pro dočasné uložení vybraných dat ze zdrojových systémů před jejich vlastní zpracováním do ostatních databázových komponent BI řešení. Jejich smyslem je urychlení výběru dat a slouží k prvotnímu ukládání netransformovaných dat z těchto systémů. DSA napomáhají:

- minimalizovat dopad transferu na výkonnost vytížených zdrojových systémů
- u systémů pracujících např. s textovými soubory (tam, kde je potřeba před vlastním zpracováním konvertovat data do databázového formátu)

DSA obsahují data, která jsou detailní, nekonzistentní, pouze aktuální, měnící se a ve stejné kultuře, v jaké jsou uložena ve zdrojových systémech (Tvrdíková, 2008).

3.1.6 Operativní úložiště dat – ODS

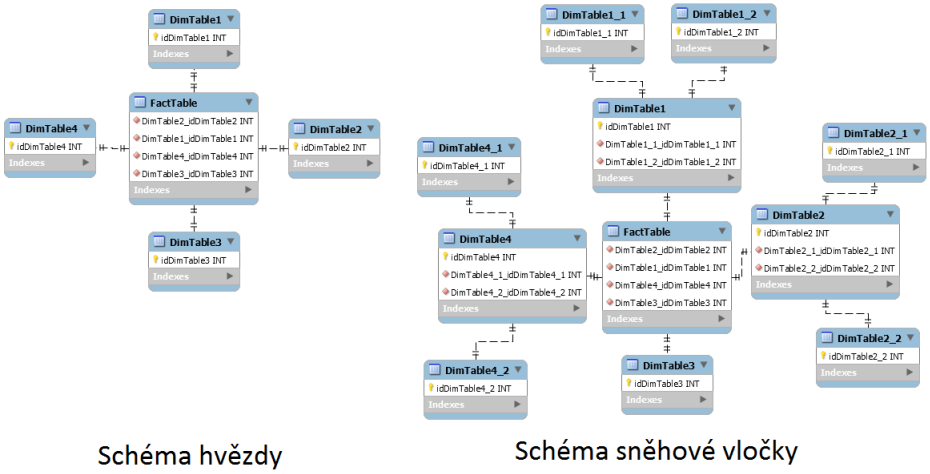
První přístup definuje ODS jako jednotné místo datové integrace aktuálních dat z primárních systémů. Jedná se o zdroj pro sledování konsolidovaných agregovaných dat s minimální dobou odezvy po zpracování (tedy sledování v téměř reálném čase). V mnoha případech takové ODS slouží jako centrální databáze základních číselníků (zákaznický, produktový) nebo pro podporu interaktivní komunikace se zákazníkem (např. podporu pracovníků call center, kdy ODS dodává aktuální konsolidovaná data o zákazníkovi, jeho profilu, použitých produktech apod.). Takto definované databáze podporují vkládání a modifikaci dat v reálném čase a jsou typicky napojeny na EAI platformy. Druhý přístup k definici ODS vymezuje operativní úložiště dat jako databázi navrženou s cílem podporovat relativně jednoduché dotazy nad malým množstvím aktuálních analytických dat. Na rozdíl od prvního přístupu, podle tohoto vymezení vzniká ODS jako derivace již existujícího datového skladu a obsahuje pouze aktuální záznamy vybraného množství dat. (Novotný, Pour, Slánský, 2005)

3.1.7 Multidimenzionální architektura

Multidimenzionální databáze jsou již optimalizované pro uložení a interaktivní využívání multidimenzionálních dat. Výhodou multidimenzionality je rychlost zpraco-

vání a efektivní analýzy multidimenzionálních dat. Základním principem, na němž jsou aplikace Business Intelligence založeny, je několikadimenzionální tabulka umožňující velmi rychle a pružně měnit jednotlivé dimenze, a nabízet tak uživateli různé pohledy na modelovanou ekonomickou realitu. Standardními dvěma dimenzemi jsou měřítka (ekonomické proměnné) a čas. Ostatní dimenze se pro jednotlivé modely definují podle potřeby, např. organizační jednotka, komodita, zákazník, dodavatel, teritorium, konkurent apod. Promítnutí všech dimenzí do jednoho bodu tvoří prvek multidimenzionální databáze. Z těchto prvků pak sestávají jednotlivé dimenze. (Novotný, Pour, Slánský, 2005)

Multidimenzionalita dat může být implementována na úrovni relační databáze a je modelována dle dvou základních schémat – viz obr. 1. V centru je vždy tzv. tabulka faktů. Obvykle do ní patří veličiny, které jsou měřitelné a měnící se v čase. Pokud je však veličina spíše diskrétní a konstantní, pak jde o naopak položku z dimenzionální tabulky, které obklopují faktové tabulky zvnějšku.



Obrázek 1: Schéma hvězdy (star) a sněhové vločky (snowflake) (Zdroj: vlastní práce)

3.1.8 OLTP, OLAP a multidimenzionální databáze

Jde o technologii založenou na multidimenzionální databázi. Hlavním principem OLAP je multidimenzionální tabulka umožňující flexibilně měnit jednotlivé dimenze a umožnit tak uživateli sledovat data týkající se ekonomické reality podniku z různých pohledů (resp. z pohledu různých zaměnitelných dimenzí). OLAP se liší od běžných transakčních systémů (OLTP) především účelem svého použití.

Zatímco běžné OLTP systémy pracují s operativními daty a mají za úkol napomáhat automatizaci a optimalizaci běžné činnosti firmy (např. ERP[3] systémy,

[3]Enterprise Resource Planning je informační systém, který integruje a automatizuje velké množství procesů souvisejících s relevantními činnostmi podniku. Typicky se jedná o výrobu, logistiku, distribuci, správu majetku, prodej, fakturaci a účetnictví.

účetnictví, personalistika), OLAP pracují s analytickými informacemi, které vznikají na základě odvození z operativních dat transakčních systémů, a jsou určeny především pro podporu rozhodovacích činností managementu. Data pro OLTP jsou nejčastěji ukládána v relačních databázích v normalizované podobě (resp. 3. normální formě). Analytická data není vhodné ukládat tímto způsobem, neboť pokud je potřeba poskytnout uživateli možnost rychle nahlížet na data z pohledu různých dimenzí, lze to v případě normalizovaně uložených dat zajistit jen velmi obtížně. Data pro OLAP jsou proto ukládána v multidimenzionální struktuře, která je optimalizována pro uložení a interaktivní zpracování (analýzy) multidimenzionálních dat, podrobněji viz např. (Thomsen, 2002). OLAP obsahuje různé úrovně agregace dat (podle hierarchické struktury definovaných dimenzí) a zahrnuje také faktor času, díky čemuž lze sledovat historický vývoj definovaných ukazatelů.

V rámci multidimenzionální analýzy, která probíhá nad OLAP kostkou, jde o to, získat hodnotu určitého vybraného ukazatele příslušejícího k uživatelem zvoleným dimenzím, viz obr. 2 (ukazatelem jsou tržby, dimenzemi čas a typ výrobku). Volbou určité kombinace dimenzí je určen prvek multidimenzionální databáze, který obsahuje hodnotu nebo algoritmus pro výpočet dané hodnoty. Standardním ukazatelem je obvykle ekonomická proměnná, která je sledována přes časovou dimenzi a současně přes několik dalších dimenzí (např. organizační jednotka, typ výrobku, zákaznické segmenty, dodavatelé, region, atd.). Dimenze jsou většinou uspořádány v hierarchické struktuře podle míry zachycovaného detailu, například dimenze týkající se výrobků může být členěna na kategorii výrobku (např. notebook) a typ výrobku nebo dimenze týkající se lokality prodeje na stát, region, apod. Systémy BI zajišťují automatické agregace hodnot (ekonomických proměnných) podle definovaných úrovní dimenzí. Pokud by totiž bylo nutné provádět součty mnoha hodnot až při zobrazování dat odpovídajících zvoleným pohledům, odezva systému by mohla být příliš velká. Předvypočítané a v OLAP kostkách uložené hodnoty agregovaných dat, odpovídající jednotlivým hierarchiím dimenzí, umožňují snadno měnit detail zobrazovaných dat a pružně zaměňovat dimenze, přes něž jsou data nahlížena. (Závodný, 2012)

3.2 Architektura BI

Obecně jsou uváděny dvě architektury řešení Business Intelligence, se kterými je možno se v podnikové praxi v současné době setkat. Patří mezi ně architektura nezávislých datových tržišť a architektura konsolidovaného datového skladu, obě jsou stručně charakterizovány v následujících odstavcích (podle Novotný, Pour, Slánský 2005).

3.2.1 Architektura nezávislých datových tržišť

Podnikové řešení Business Intelligence je tvořeno několika nezávislými datovými tržišti, která slouží pro potřeby jednotlivých specifických útvarů podniku. Každé

tržiště zpravidla zahrnuje veškeré komponenty BI, které umožňují získat, transformovat, ukládat a prezentovat analytická data uživatelům. Ačkoli jsou jednotlivá tržiště relativně nezávislá, je snahou podniků vzájemně je propojit přes tzv. sdílené dimenze, tedy dimenzionální tabulky, které jsou opakovaně použity v různých datových tržištích. Celkové řešení BI je v rámci této architektury obvykle budováno postupně, přičemž každé nově vytvářené tržiště má za cíl využít co nejvíce již existujících dimenzí. Sdílené dimenze (např. zákazník, produkt, atd.) zajišťují vzájemnou konzistenci reportingu jednotlivých tržišť. Tato architektura je využívána zejména pokud je potřeba pokrýt analytické potřeby jednotlivých oddělení podniku co nejrychleji, při relativně nižší ceně projektu, přičemž není příliš kladen důraz na budoucí integraci řešení, neboť určitým nedostatkem této architektury je obtížnější integrace jednotlivých datových tržišť do celopodnikového řešení. To je dáno zejména náročnou implementací sdílených dimenzí a potřebou vytvořit jednotnou vrstvu reportingu nad několika datovými tržišti. (Závodný, 2012)

3.2.2 Architektura konsolidovaného datového skladu

V rámci této architektury jde primárně o vybudování integrovaného BI řešení. Jeho základem je konsolidovaný datový sklad obsahující jak detailní, tak agregovaná data. Řešení je doplněno o závislá datová tržiště, která využívají konsolidovaná data datového skladu. Vytvoření této architektury vyžaduje detailní počáteční analýzu požadavků a vytvoření celkové koncepce, i proto bývá zejména v počátečních fázích časově i finančně náročnější. Na druhé straně přináší výhodu v podobě konsolidovaného reportingu, snazší rozšiřitelnosti bez nutnosti řešit náročnější integrační problémy a větší podporu analytických a dataminingových úloh. Tato architektura bývá zaváděna buď jednorázově, zejména pokud jde o menší projekty, nebo přírůstkově, pokud jde o větší časově náročnější řešení.

4 BI Sémantický model (BISM)

Kapitola se věnuje podstatě sémantického modelu, popisuje tabulární a multidimenzionální modelování, jejich slabé a silné stránky a také tomu, kdy by měla být v projektu využita jedna, či druhá varianta. Blíže také seznamuje čtenáře s nástroji BI vytvořené firmou Microsoft a zkoumá podstatu evoluce jejich analytických služeb.

4.1 Definice BISM

Termín BI Semantic Model, zkráceně BISM, lze najít v mnoha odborných knihách, ale také na internetových diskuzích a fórech. Tento termín je oficiálně podporován i firmou Microsoft (Microsoft, BISM white paper, 2014). BISM neindikuje, zda je použit multidimenzionální nebo tabulární model, nýbrž místo toho popisuje funkci SSAS v balíčku Microsoft BI. Jedná se o sémantickou vrstvu nad datovým skladem, která přidává bohatou vrstvu metadat[4], která obsahuje hierarchie, měřítka a kalkulace. Z tohoto pohledu je velmi podobný termínu Unified Dimensional Modeling (UDM), který byl užíván v době vydání SQL Serveru 2005. Někdy se lze však setkat s případy, kdy BISM nesprávně odkazuje pouze na tabulární model.

4.2 Microsoft BI ekosystém

V Microsoft ekosystému BI nepředstavuje pouze jeden produkt, ale sestává z mnoha funkčních celků distribuovaných po jednotlivých produktech.

4.2.1 Analysis Services

Pod SSAS, celým názvem také SQL Server Analysis Services, rozumíme OLAP databázi, která je vysoce optimalizovaná pro dotazy, které se běžně vyskytují v BI prostředí. Provádí mnoho operací, které jsou totožné s operacemi u relačních databází, ale také se od relační databáze v mnohém liší. Ve většině případů je snazší vytvořit BI řešení za pomoci Analysis Services v kombinaci s relační databází, jako je SQL Server, než pouze za pomoci SQL Serveru. Analysis Services rozhodně nenahrazují relační databázi, nebo správně navržený datový sklad.

Na SSAS lze pohlížet jako na speciální vrstvu metadat nad datovým skladem. Tato vrstva obsahuje informace o tom, jak mají být spojeny faktové a dimenzionální tabulky, jak mají být agregovány ukazatele (measures), jaké má uživatel možnosti proházet daty pomocí hierarchií a podobně. Tato vrstva pak zahrnuje modely, které obsahují business logiku datového skladu. Koncový uživatel pak dotazuje tyto modely namísto nízkoúrovňové relační databáze. Dotazy, které uživatelé píší, jsou následně mnohem jednodušší, neboť stačí pouze specifikovat, které řádky a sloupce jsou nezbytné a model pak vhodně aplikuje business logiku, aby zajistila, že vrácená čísla dávají smysl. Důležitý je také fakt, že není možné napsat dotaz tak, aby

[4]strukturovaná data o datech

vracel nesprávné výsledky kvůli chybě koncového uživatele, jako je špatné spojování tabulek, či sumarizování sloupců, u kterých to nedává smysl. To dále také znamená, že reportovací a analytické nástroje pro koncové uživatele mají méně práce se zpracováním dotazů a poskytováním zřetelného vizuálního rozhraní pro tvorbu těchto dotazů.

Na SSAS lze také pohlížet jako na typ cache[5], která slouží ke zrychlení reportingu. Většinou je SSAS načteno s kopií dat v datovém skladu. Následně jsou všechny reportovací a analytické dotazy prováděny nad SSAS namísto relační databáze. I přes to, že moderní relační databáze jsou vysoce optimalizovány a obsahují mnoho prvků zaměřené přímo na BI reporting, SSAS je určena přímo pro tento druh zátěže a ve většině případů dosahuje mnohem lepších výsledků. Pro koncového uživatele je rychlost zpracování dotazů velice důležitá, protože mu umožňuje pracovat zároveň s jeho tokem myšlenek bez přerušení. Co se týče IT oddělení, jeho největší výhoda je, že může vytváření reportů přenechat koncovému uživateli. Klasický problém v BI projektech, které nevyužívají OLAP je ten, že IT oddělení musí vytvořit nejen datový sklad, ale k nim i soubor reportů. Tím se zvyšuje množství času a úsilí a může být příčnou frustrace pro business, když zjistí, že IT není schopno porozumět jeho požadavkům na reporting, nebo tento reporting neposkytuje dost rychle. Pokud je použita OLAP databáze jako je SSAS, IT oddělení může vystavit modely, které obsahuje, koncovým uživatelům a dát jim tak možnost, aby si vytvořili takové reporty, které vyhovují jejich potřebám a za použití takového prostředku, který jim vyhovuje. Nejznámější klientský nástroj je Microsoft Excel. Již od roku 2000 jsou PowerPivot tabulky schopny připojit se přímo k SSAS kostkám a nejnovější verze Excelu obsahují velmi výkonné funkce jakožto klientský nástroj pro SSAS. (Russo, Ferrari, Webb 2012)

4.2.2 Microsoft BI nástroje

Jak uvádí autoři (Russo, Ferrari, Webb 2012), můžeme nástroje BI od firmy Microsoft rozdělit na dvě skupiny. Jedna skupina představuje ty produkty, které jsou využívány jako přímá součást, případně v kooperaci s SQL Server 2012, a to:

- **SQL Server relační databáze**
  Platforma pro relační datový sklad.
- **SQL Azure**
  SQL Server běžící na cloudu, momentálně nepříliš využívaný pro BI.
- **Paralelní datový sklad**
  Vysoce specializovaná verze SQL serveru, zaměřená na podnikové datové sklady, které musí pojmout několik terabytů dat, přičemž je schopná rozprostřít zátěž na několik serverů.

[5]česky též keš nebo mezipaměť je označení pro vyrovnávací paměť používanou ve výpočetní technice

- **SQL Server Integration Services**
  ETL nástroj, tedy nástroj pro extrakci dat z externích zdrojů, jejich transformaci do vhodné formy a následné loadování, neboli načtení dat do datového skladu.
- **Hadoop**
  Nejrozšířenější open-source nástroj pro agregaci a analýzu objemných dat.
- **SQL Server Reporting Services**
  Zkratka SSRS značí nástroj, který slouží k vytváření statických a semistatických reportů a pravděpodobně nejpoužívanější nástroj z SQL server balíčku.
- **SQL Azure Reporting**
  Reportovací služby SQL Serveru na cloudu.
- **PowerView**
  Nástroj Power View je součástí doplňku služby SQL Server 2012 Reporting Services pro server Microsoft SharePoint Server 2010 Enterprise Edition a jedná se o prostředí, které slouží k interaktivnímu zkoumání dat a vytváření vizualizací a prezentací. Tento nástroj poskytuje podnikovým uživatelům, jako jsou například analytici dat, pracovníci s rozhodovací pravomocí a pracovníci pracující s informacemi, možnost intuitivního vykazování ad hoc. Uživatelé mohou snadno vytvořit a interagovat se zobrazením dat z datových modelů založených na sešitech PowerPivot, které jsou publikovány v galerii PowerPivot, nebo tabulkovými modely nasazenými v instancích služby SQL Server 2012 Analysis Services (SSAS). Nástroj Power View je aplikace Silverlight založená na prohlížeči, která se spouští ze serveru SharePoint Server 2010 a umožňuje uživatelům prezentovat nápady a sdílet je s kolegy v organizaci prostřednictvím interaktivních prezentací. (Microsoft, 2014)
- **StreamInsight**
  Event-processing platforma pro analýzu dat, která vstupují do relační databáze příliš rychle a ve velkém objemu.
- **Master Data Services**
  Nástroj k tzv. správě kmenových dat.
- **Data Quality Services**
  Nástroj pro čistění dat a zajištění jejich kvality.
- **PowerPivot**
  Samoobslužný nástroj, který umožňuje koncovému uživateli vytvořit vlastní sadu reportů v programu Excel a publikovat je na SharePoint serveru. Více informací o tomto nástroji lze nalézt v dalších kapitolách.

Druhou skupinou nástrojů jsou pak ty obsažené v MS Office balíčku.

- **SharePoint**
  Je webový aplikační framework a platforma, která slouží organizacím pro vytváření a sdílení webového obsahu.
- **PerformancePoint Services**
  Nástroj pro tvorbu BI dashboardů na SharePoint serveru.
- **Excel**
  Velmi známý tabulkový procesor, ale také pravděpodobně, ve spojení s PowerPivot, nejpoužívanější BI nástroj na světě.

4.2.3 Samoobslužné a korporátní BI

Jeden z nejvýraznějších trendů poslední doby v BI průmyslu jsou takzvané samoobslužné BI nástroje, jako je QlikView či Tableau. Tyto nástroje mají za cíl poskytnout pokročilým uživatelům možnost vytvářet BI řešení malého rozsahu s malou pomocí IT oddělení, či úplně bez ní. Oproti tomu v historii dominovalo klasické korporátní BI, kdy je firmě poskytnuto komplexní řešení striktně kontrolované IT oddělením. Ve firmách menšího rozsahu je však takové řešení často neefektivní a firma si ho často ani nemůže dovolit. V takovém prostředí se proto čím dál častěji využívá samoobslužné BI, jakožto levnější, agilnější a méně riskantní alternativa.

Samoobslužné BI však není „samospásné“. Přesto, že se jedná o agilní vývoj BI řešení, který je zaměřený na business a má bleskové odezvy, má i své slabé stránky. Může totiž zvýraznit problémy spojené se zachováváním zastaralých údajů, špatnou kvalitou dat, či nedostatkem integrace mezi různými zdroji dat.

V jistém smyslu bylo SSAS vždy také z části samoobslužným BI nástrojem, neboť umožňuje koncovým uživatelům vytvářet vlastní dotazy a reporty, ale stále vyžaduje IT odborníka k navržení a vybudování SSAS databáze nad datovým skladem. Jako reakci na rychle rostoucí poptávku trhu po samoobslužném řešení vydala společnost Microsoft vlastní samooblužný BI nástroj zvaný PowerPivot. (Jorgensen, 2012)

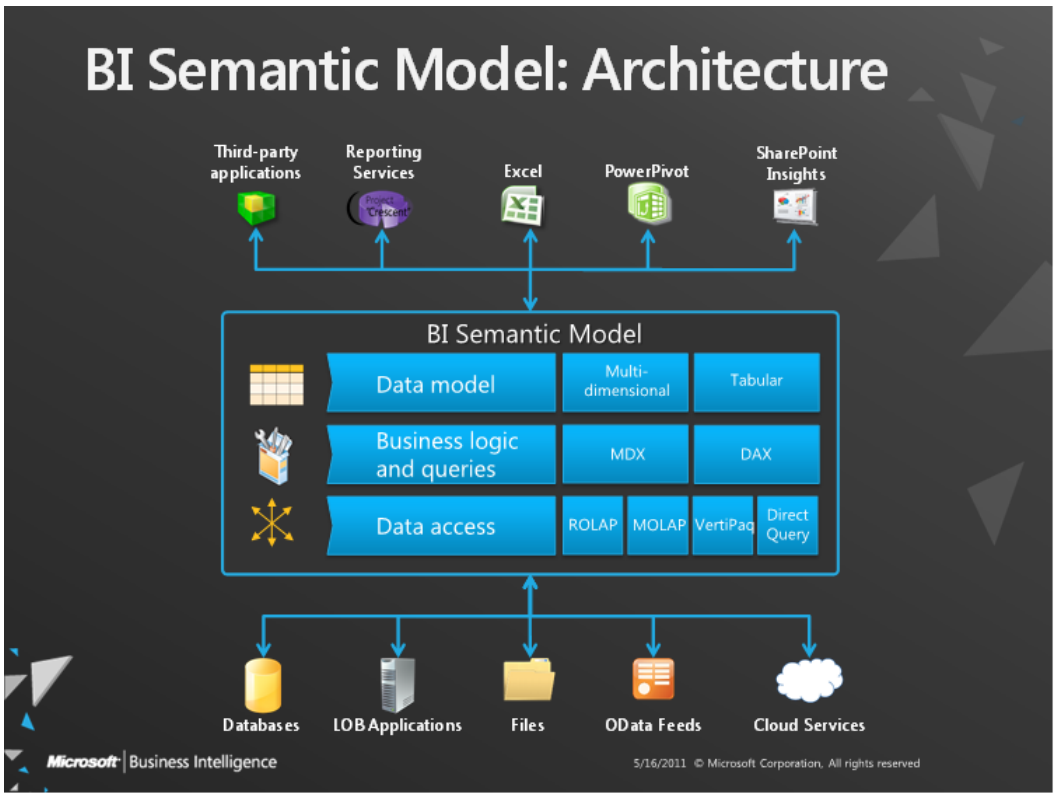
4.2.4 PowerPivot pro Excel a SharePoint

Microsoft PowerPivot je výkonný samoobslužný BI nástroj pro ad-hoc analýzu extrémně velkých datových vzorků, který je součástí známého prostředí Excelu. Lze provádět analýzy bez použití serveru, analýzu hloubkovou, ve stylu OLAP, sloupcovou analýzu, jakož i vytvořit si vlastní sloupcové a tabulkové kalkulace. PowerPivot verze 1 byla nejen součástí SQL Serveru 2008 R2, ale zároveň byla dodávána jako doplněk pro Excel 2010 a nové serverové služby pro SharePoint 2010. Umožnuje rychle analyzovat miliony řádků bez nutnosti datového skladu, odborníka na BI, nebo serverové infrastruktury. Cílem tohoto produktu je umožnit uživatelům nezávisle vytvořit BI řešení, které čerpá data z různých zdrojů, jako je SQL Server, SSAS, či obyčejná excelovská tabulka, nebo tabulka někde na webu. Na rozdíl od tradičního Unified Dimensional Modelu, který zpracovává kostku a ukládá ji na

disk, PowerPivot využívá sloupcové komprese, uložení v RAM paměti a tabulárního modelu, o kterém pojednává další kapitola.

4.3 Architektura Analysis Services 2012

Na SSAS 2012 se dá pohlížet jako na balíček dvou produktů v jednom. Jeho architektura se totiž dělí na dva modely (viz obr. 2). První z nich je tradiční model, který je součástí SSAS již od jeho prvních verzí, a který se nyní nazývá multidimenzionální. U tohoto modelu s novou verzí nepřišlo mnoho změn a je tedy téměř totožný s multidimenzionálním modelem u předchozí verze. Druhý model, tabulární, je však novinkou a právě tímto modelem se především zabývá tato diplomová práce.



Obrázek 2: BI Sémantický model (Microsoft, 2011)

4.4 Tabulární model

Databáze je tvořena jednou, či více tabulkami. Tabulka v tabulárním modelu (někdy také v české literatuře zvaný tabelární) je velmi podobná tabulce v relační databázi. Obvykle je načtena z jedné tabulky v relační databázi, nebo je výsledkem SQL SELECT dotazu. Tabulka má pevný počet sloupců, které jsou definovány v čase návrhu a mohou mít proměnný počet řádků v závislosti na množství načtených dat.

Každý sloupec má pevný datový typ, tedy množina povolených hodnot je například pouze celá čísla, či znaky.

Je také možné definovat relace mezi tabulkami v čase návrhu. Narozdíl od SQL není možné definovat tyto relace v době dotazování. Všechny dotazy mohou využít pouze předdefinované relace. Tyto relace mezi tabulkami mohou však být označeny jako aktivní či neaktivní, čímž je možno simulovat efekt relací, které neexistují v dotazech a kalkulacích. Všechny relace mají kardinalitu 1:N a musí zahrnovat pouze jeden sloupec z každé ze dvou tabulek. Není možné definovat relace, které jsou explicitně 1:1, či M:N. Nicméně lze dosáhnout stejného výsledku tvořením specifických dotazů a kalkulací. Není možné ani navrhnout relace, které jsou založeny na více než jednom sloupci tabulky nebo rekurzivní relace, které spojují příkazem JOIN tabulku do sebe samé. (Jorgensen, 2012)

Tabulární model využívá engine, který je založený čistě na ukládání dat do operační paměti, přičemž ukládá pouze kopii dat na disk, aby tak nedošlo ke ztrátě dat, pokud je služba restartována. Zatímco multidimenzionální model, jako většina relačně databázových enginů, ukládá data ve formě řádků, tabulární model využívá sloupcově orientovanou databázi zvanou „xVelocity in-memory analytics engine“, který ve většině případů znamená významné vylepšení výkonu dotazu.

Dotazy a kalkulace v Tabularu jsou definovány v „Data Analysis eXpressions“ (DAX), což je nativní dotazovací jazyk tabulárního modelu a PowerPivotu. Klientské nástroje, jako je PowerView, mohou generovat DAX dotazy pro získání dat z tabulárního modelu, nebo je možno psát vlastní DAX dotazy a využít je v reportech. Je také možné napsat dotazy za použití jazyka MDX a multidimenzionálního modelování. To znamená, že tabulární model je zpětně kompatibilní s mnohými klientskými nástroji SSAS, jako je Excel či SSRS a nástroji třetích stran.

Odvozené sloupce (v SSAS zvané „calculated columns“) mohou být vloženy do tabulky v tabulárním modelu. Využívají přitom DAX výrazů, které vracejí hodnoty na základě již načtených dat v ostatních sloupcích stejné, či jiné tabulky ve stejné SSAS databázi. Odvozené sloupce jsou naplněny až v čase zpracování, po němž se chovají naprosto stejně jako klasické sloupce. Měřítka (measures) mohou být také definována v tabulkách za pomoci DAX výrazů. Na měřítko lze pohlížet jako na DAX výraz, který vrací agregovanou hodnotu na základě dat v jednom či více sloupcích. Jednoduchý příklad měřítka by mohla být suma všech hodnot ve sloupci, která obsahuje objem prodejů. „Key performance indicators (KPI)“ jsou indikátory velmi podobné měřítkům, ale jedná se o soubor kalkulací, který umožňuje určit, jak dobře si měřítko vede v porovnání s plánovanou hodnotou a zda se jí časem přibližuje.

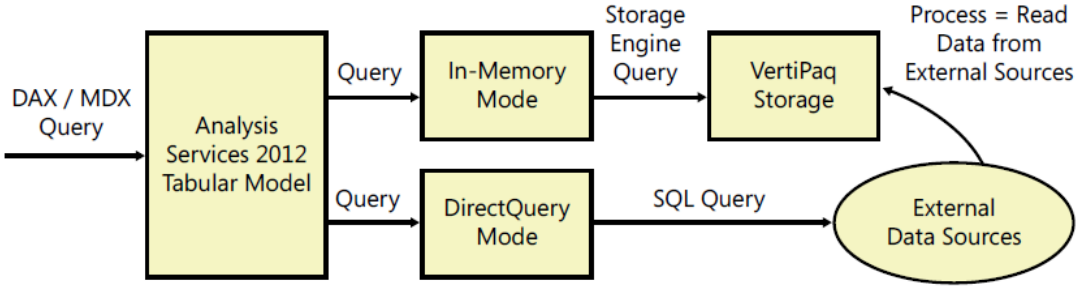
Většina front-endových nástrojů, jako je například Excel, používá nějakou formu kontingenčních tabulek pro dotazování tabulárního modelu. Sloupce z různých tabulek mohou být přetaženy do os řádků a sloupců kontingenční tabulky. Unikátní hodnoty z těchto sloupců se pak stávají jednotlivými řádky a sloupci kontingenční tabulky. Měřítka ukazují agregované numerické hodnoty uvnitř tabulky. Celkový výsledek je podobný SQL Group By dotazu, přičemž definice toho, jak se mají data agregovat, se nachází uvnitř měřítka a nemusí být součástí dotazu. Aby bylo

možné dosáhnout ještě přehlednějších výsledků, je možno definovat hierarchie u tabulek tabulárního modelu, které vytvoří předdefinované, několikaúrovňové „drilovací“[6] cesty. Perspektivy[7] mohou skrýt určité části komplexního modelu, což může zlepšit použitelnost a bezpečnostní role mohou být použity k zamezení přístupu určitých uživatelů k určitým řádkům tabulky. Samotné perspektivy však neslouží pro zabezpečení, neboť i přesto, že jsou určitá data skryta, mohou být opět zobrazena například MDX dotazy. (Lidberg, 2013)

4.4.1 xVelocity a DirectQuery

U tabulárního modelu je možno zvolit mezi dvěma módy nasazení. Jedná se o In-Memory a DirectQuery. In-Memory mód je také zvaný xVelocity in-memory analytics engine (dříve VertiPaq), který představuje interní engine pro ukládání dat (storage engine) použitý v tomto módu. DirectQuery se spoléhá na externí datový zdroj a konvertuje dotaz zaslaný SSAS na SQL dotaz zaslaný externímu datovému zdroji.

Na obrázku 3 je vidět, co se stane, když je dotaz zaslán SSAS 2012 tabulárnímu modelu. Dotaz je zpracován jedním ze dvou postupů, v závoslosti na módu nasazení. U In-Memory módu jsou data získána přístupem do xVelocity enginu, kdežto u DirectQuery módu je dotaz převeden na SQL a zaslán externímu datovému zdroji bez použití úložiště, nebo cache v SSAS.



Obrázek 3: Diagram vykonání dotazu v SSAS Tabular (Russo, Ferrari, Webb 2012).

In-Memory mód je výchozí pro nasazení tabulárního modelu, je to tentýž engine, který používá PowerPivot pro Excel. In-Memory mód používá storage engine zvaný xVelocity in-memory analytics engine – sloupcový storage engine pro ukládání dat, který vykazuje nejlepší výkon pro dotazování, přičemž požaduje, aby data byla zpracována a uložena v SSAS paměti.

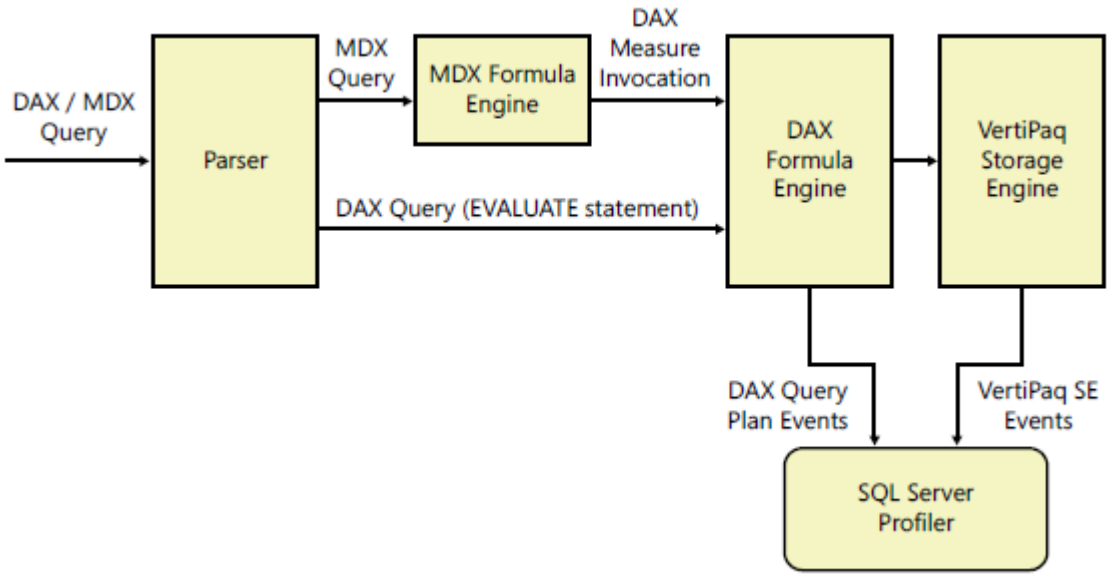
[6]Drill-down, neboli drilování je v širším smyslu jednoduchou technikou umožňující rozpadnout řešený problém na jednodušší části, se kterými si lze lépe poradit. Většinou se jedná o zobrazení určité agregované hodnoty ve větším detailu, aby bylo jasné, z čeho tato hodnota vychází.

[7]Perspektivy vytvořené v SSAS umožňují zobrazit uživateli soubor dat nad kostkou, který se jeví jako samostatná kostka, nicméně se jedná pouze o předdefinovaný soubor objektů z určité kostky, který slouží pro redukci a zpřehlednění zpravidla velkých objemů dat.

In-Memory mód je sice obvykle favorizovaná možnost u tabulárních modelů, ale stále zůstává možnost využít DirectQuery, která využívá externí zdroj dat jakožto jediný repozitář bez duplikování dat v SSAS. Použitím DirectQuery se tabulární model stává pouze sémantickou vrstvou nad SQL Server databází a každý DAX dotaz na tabulární model je převeden na SQL dotaz. (MDX dotazy nejsou v módu DirectQuery podporovány). To zahrnuje DAX kalkulace, které jsou přeloženy do ekvivalentních SQL výrazů. DirectQuery také umožňuje real-time dotazování a nevyžaduje kopii dat uloženou v SSAS. Má ale také několik omezení a nevýhod. (Dewald, Hughes, Turley, 2013)

Podobnou volbu můžeme pozorovat i u multidimenzionálního modelu mezi režimy MOLAP a ROLAP, o kterých se lze dozvědět víc v následující kapitole. V multidimenzionálním modelu však můžeme tuto volbu udělat na úrovni partitionů, kdežto u Tabularu je třeba se rozhodnout na úrovni celého modelu.

Storage engine xVelocity je dotazován na nižší úrovni plánu spuštění dotazu (query execution plan) ve chvíli, kdy musí být data vypočtena. Na obrázku 4 lze vidět, jak je dotaz zpracován, pokud je mód In-Memory aktivní.



Obrázek 4: Diagram vykonání dotazu za použití In-Memory módu (Russo, Ferrari, Webb 2012).

S DAX a MDX dotazy zaslanými tabulárnímu modelu je nakládáno odlišně. DAX dotaz je vyhodnocen pomocí DAX formula enginu, který vygeneruje DAX query plan, který je pak převeden na dotazy zaslané xVelocity storage enginu. MDX dotaz je analyzován za pomoci MDX formula enginu, který volá DAX formula engine, aby mu pohl s řešením DAX měřítek a generuje query plan, který zpracovává požadavky na xVelocity storage engine. To znamená, že MDX dotaz není převeden na ekvivalentní DAX syntax, ale přímo generuje jeden či více dotazů na DAX for-

mula engine a následně xVelocity pro získání hodnot a vyhodnocení DAX měřítek. (Dewald, Hughes, Turley, 2013)

Je důležité vědět, že MDX i DAX formula enginy provádějí jednovláknové operace, kdežto xVelocity storage engine umožňuje využití více jader. Z toho důvodu je vhodné psát dotazy, které maximalizují výkon za pomoci využití nejmenšího množství požadavků na xVelocity, jelikož vytížení formula enginu bude pak nízké.

4.4.2 Řádková vs. Sloupcová databáze

Většina relačních databází, včetně SQL Serveru, jsou řádkově orientované databáze. To znamená, že veškerá tabulková data v databázi jsou ukládána do řádků. Vezměme v úvahu následující tabulku.

Tabulka 1: Data uložená v řádkové databázi

ID Studenta	**Jmeno**	**Prijmeni**	**Obor**	**Rocnik**
1	Jindrich	Zeleny	Informatika	5
2	Rudolf	Cerveny	Ekonomie	3
3	Karel	Modry	Matematika	5

Řádkově orientovaná databáze fyzicky ukládá data řádek po řádku. Je obvyklé, že je vytvořen index, který ukazuje na všechny sloupce určitého řádku, jako například v následující tabulce.

Tabulka 2: Indexování dat v řádkové databázi

ID řádku	**Data**
1	1,Jindrich,Zeleny,Informatika,5
2	2,Rudolf,Cerveny,Ekonomie,3
3	3,Karel,Modry,Matematika,5

Fyzická implementace závisí na databázovém produktu. Například SQL Server dělí úložný prostor do stránek (po 8 kB) a každá stránka obsahuje jeden nebo více řádků. Čím je řádek delší, tím menší je počet řádků na stránce. K načtení řádku databáze musí načíst nejdřív stránku, která řádek obsahuje. Pro zvýšení výkonnosti SQL Server obvykle používá indexy, které jsou implementovány uložením většího počtu kratších řádků v nižším počtu stránek. Tyto stránky jsou seřazeny tak, aby dále redukovaly množství stránek nutných k vykonání dotazu. Obecně řádkově orientovaná databáze potřebuje úplné prohledání všech řádků tabulky (tzv. full table scan), pokud dotaz pracuje se všemi hodnotami z nějakého sloupce v tabulce (například agregace). Časové náklady na kompletní scan tabulky jsou stejné nehledě na počet dotazovaných sloupců, není-li předem vytvořen určitý index.

Sloupcová databáze využívá jiný přístup. Namísto toho, aby považovala řádek tabulky za hlavní úložnou jednotku, považuje každý sloupec za samostatnou entitu a ukládá data zvlášť pro každý z nich. Data mohou být například uložena následovně:

Tabulka 3: Data uložená ve sloupcové databázi

Název sloupce	**Data**
ID Studenta	1,2,3
Jmeno	Jindrich,Rudolf,Karel
Prijmeni	Zeleny,Cerveny,Modry
Obor	Informatika,Ekonomie,Matematika
Rocnik	5,3,5

Tato technika umožňuje velmi rychlé dotazování na data z jediného sloupce, ale znamená vyšší výpočetní nároky při zpracování dotazů na více položek z jednoho řádku. Nejhorší možný scénář je požadavek na všechny položky řádku, který vyžaduje přístup do všech řádků.

Z pohledu načtení dat by mohla být sloupcově orientovaná databáze rychlejší, neboť přístup k datům je optimalizovaný pro mnoho dotazovacích scénářů. Nejčastější požadavky v analytickém systému vyžadují data pouze z několika sloupců obvykle k agregaci dat ze sloupce pomocí seskupování dle hodnot v jiných sloupcích. Nicméně jakmile je potřeba přidat nebo odebrat řádek tabulky, je nutno přidat nebo odebrat položku v každém sloupci. To je obvykle mnohem dražší operace, než v řádkově orientované databázi. Přidávání, mazaní, či změnu řádku navíc není možné provést za pomoci xVelocity enginu.[8] (Russo, Ferrari, Webb 2012)

4.4.3 xVelocity engine

xVelocity je velmi rychlý engine, který ukládá vysoce komprimovaná data. To je umožněno specifickou architekturou ukládání. Každá zpracovaná tabulka je rozdělena do sloupců a každý sloupec má svůj úložný prostor. Pro každý sloupec je vytvořen slovník všech unikátních hodnot, to znamená, že tento prostor je v podstatě bitmapový index, který odkazuje na slovník. Slovník i bitmapový index následně prochází kompresí ve vysokém poměru a takto se ukládají do paměti RAM i na disk. Uložení dat na disku slouží jako záloha, díky které databáze nemusí byt opětovně zpracována v případě restartování služby. Všechny dotazy jsou provedeny nad daty, která jsou načtena do paměti. Vezměme v úvahu následující tabulku:

[8] Některé jiné sloupcově orientované databáze operace insert, delete a update na úrovni jednotlivých řádků podporují. Uvedená limitace je specifická pro současnou verzi xVelocity enginu.

Tabulka 4: Tabulková data

ID	**Kategorie**	**Produkt**	**Barva**
546	Prislusenstvi	Zamek	Stribrna
235	Prislusenstvi	Brzdy	Stribrna
987	Kola	Horske kolo	Cervena
988	Kola	Horske kolo	Stribrna
742	Kola	Cestovni kolo	Cervena
744	Kola	Cestovni kolo	Stribrna

xVelocity ukládá tuto tabulku následovně. Pro každý sloupec je vytvořen seřazený slovník všech unikátních hodnot a bitmapový index odkazuje aktuální hodnoty každé položky ve sloupci do slovníku za využití od indexu začínajícího od nuly.

Tabulka 5: Uložení dat v xVelocity enginu

Sloupec	**Slovnik**	**Hodnoty**
ID	235,546,742,744,987,988	1,0,4,5,2,3
Kategorie	Prislusenstvi, Kola	0,0,1,1,1,1
Produkt	Brzdy, Zamek, Horske kolo, Cestovni kolo	1,0,2,2,3,3
Barva	Cervena, Stribrna	1,1,0,1,0,1

Lze vypozorovat, že prostor potřebný k uložení slovníku může být nejdražší částí, zvláště v případech, kdy mají sloupce mnoho unikátních hodnot. Sloupce, které obsahují menší množství unikátních hodnot potřebují menší slovník, což má za následek efektivnější bitmapové indexování. xVelocity poskytuje několik optimalizačních metod, kterými lze rychle vybudovat a skenovat tyto struktury, avšak obecný princip toho, co se děje v zákulisí, zůstává stejný. Pokud je vybrán tento model zpracování dat, je třeba mít na paměti dvě implikace, které z této volby vycházejí:

- Aby bylo možné komprimovat data s vysokou efektivností, kompletní tabulka (nebo přinejmenším její partition) musí být zpracována, jako jeden celek. Sestavení slovníku se všemi unikátními hodnotami je kritické.
  Veškerá data jsou uložena uvnitř SSAS na disk.
- Operace INSERT, UPDATE a DELETE představují vysoké výpočetní nároky a nejsou podporovány v SSAS 2012

xVelocity je také „in-memory“ databáze. To znamená, že byla navržena a optimalizována pro situace, kdy je databáze kompletně načtena do paměti. Kvůli uložení většího množství dat a vylepšení výkonu, jsou data také komprimována do paměti a dynamicky dekomprimovaná během každého dotazu. Proto jsou kladeny větší nároky na hardwarové vybavení, a to především rychlé procesory a vysokou propust-

nost paměti. SSAS zvládne stránkovat data na disk, což by mělo být ale omezeno pouze na scénáře, kdy je stránkování dočasnou aktivitou. (Russo, Ferrari, Webb 2012)

4.5 Multidimenzionální model

Na nejvyšší úrovni je multidimenzionální model (MDM) velmi podobný tabulárnímu. Data jsou uložena v databázi a databáze jsou navrhovány v SQL Server Data Tools (SSDT, dříve také Business Intelligence Development Studio, či BIDS) a spravovány v SQL Server Management Studiu (SSMS).

Rozdíly začínají být patrné na databázové úrovni, kde převažují multidimenzionální koncepty nad těmi relačními. V multidimenzionálním modelu jsou data modelována jako množina kostek a dimenzí, nikoliv jako tabulek. Každá kostka je tvořena jednou nebo více skupinami měřítek (measure groups) a každé měřítko v kostce je obvykle mapováno na jedinou faktovou[9] tabulku v datovém skladu. Skupina měřítek obsahuje jedno či více měřítek, která jsou velmi podobná těm v tabulárním modelu. Kostka pak obsahuje dvě či více dimenzí. Jednu speciální dimenzi, která obsahuje všechna měřítka z každé ze skupin měřítek a dále různé další dimenze, jako je čas, produkt, geografie, zákazník, které se mapují na dimenze logického modelu. Každá z těchto klasických dimenzí obsahuje jeden nebo více atributů a ty mohou být použity jakožto jednoúrovňové hierarchie či k vytvoření víceúrovňových uživatelských hierarchií. Hierarchie pak mohou být použity k tvoření dotazů. Uživatelé začnou analyzovat data na vysoce agregované úrovni, jakou je například rok v časové dimenzi a poté se mohou dostat do nižších úrovní, jako je čtvrtletí, měsíc a den. (Warren, 2013)

Protože je multidimenzionální model přímým nástupcem předchozích verzí analytických služeb SQL studia, má velmi širokou a pokročilou paletu možností reprezentující více než dekádu vývoje i přes to, že některé z nich nejsou využívány příliš často. Většina prvků dostupných v tabulárním modelu je dostupná i v multidimenzionálním modelu, avšak multidimenzionální model má navíc mnoho prvků, které zatím nebyly implementovány v Tabularu. Z pohledu uložení dat, multidimenzionální model ukládá data třemi způsoby (Sabherwal, 2010):

- **Multidimenzionální OLAP (MOLAP)**
  Veškerá data jsou uložena uvnitř SSAS na disk.
- **Relační OLAP (ROLAP)**
  SSAS se chová čistě jako vrstva metadat, přičemž žádná data nejsou uložena v SSAS. Při dotazování na data z kostky se SQL dotazy provádí přímo nad relační databází.

[9] Faktová tabulka obvykle obsahuje vysoký objem numerických dat, která se analyzují, agregují, třídí, a podobně.

- **Hybridní OLAP (HOLAP)**
  Funguje stejně jako ROLAP, avšak v SSAS jsou uloženy některé preagregované hodnoty.

V drtivé většině implementací převažuje MOLAP, ROLAP je někdy využíván pro tzv. „real-time BI“ a HOLAP se nevyužívá téměř nikdy.

Podstatná odlišnost mezi multidimenzionálním a tabulárním modelem spočívá v použití odlišných dotazovacích jazyků a typu provádění kalkulací, které podporují. Nativní jazyk multidimenzionálního modelu je MDX a jedná se o jediný jazyk pro definici dotazů a kalkulací. Jazyk MDX zaznamenal úspěch a je podporován mnoha nástroji třetích stran pro SSAS. Byl také stanoven jako standard společenstvím „XMLA Council“, čímž byl ve výsledku osvojen také mnohými dalšími OLAP nástroji, které jsou přímými soupeři SSAS. Přesto u jazyka MDX často i mezi mnohými BI profesionály nastává problém, a ten spočívá v pomalu stoupající křivce učení, neboť i přesto, že se jedná o velmi mocný nástroj, tak koncepty, které využívá, jako jsou dimenze a hierarchie, jsou velmi odlišné od toho, na co si zvykli při používání jazyka SQL.

4.6 Srovnání prvků obou modelů

Při rozhodování o správném modelu je na místě také vzít v potaz, jaké prvky funkcionality konkrétní model nabízí. Ne každý prvek je důležitý a potřebný u všech projektů. Také lze často v Tabulárním modelu nahradit ekvivalentní chybějící funkcionalitu MDM sofistikovanými DAX dotazy. V tabulárním modelu chybí oproti multidimenzionálnímu několik prvků. Seznam těchto prvků je následující (Feldmann, 2013):

- **Writeback**
  Možnost koncového uživatele zapisovat hodnoty zpět do multidimenzionální databáze.
- **Překlady**
  Metadata se v MDM zobrazují uživatelům v závislosti na lokalizaci v různých jazycích.
- **Bezpečnost měřítek dimenzí**
  Kdy přístup k jednotlivým měřítkům může být odepřen.
- **Bezpečnost buněk**
  Přístup do jednotlivých buněk může být povolen či odepřen.
- ***Ragged* hierarchie**
  Český ekvivalent by mohl znít roztrhané hierarchie. V podstatě jde o vynechané uzly, kdy například v hierarchii kontinent-stát-region-město vynecháme uzel region.

- ***Role-playing* dimenze**
  Jakmile je dimenze jednou navržena a zpracována, může vystupovat pod různými jmény a pracovat s různými relacemi. Tuto funkcionalitu sice lze v tabulárním modelu napodobit, ale s rozdílem, že pokud je žádoucí tutéž tabulku vidět na dvou místech naráz, tak je třeba ji načíst do paměti dvakrát, což může způsobit nárůst času potřebného ke zpracování a údržbě.

- ***Scoped assignments*, unární operátory**
  Tyto pokročilé možnosti kalkulací, které jsou v MDM dostupné, jsou využívány především ve finančních aplikacích. Lze tedy říci, že kvůli absenci těchto možností společně s absencí funkce writeback a hierarchie rodič/potomek není Tabular příliš vhodný pro finanční typ aplikace.

Za funkcionalitu, která je podporována pouze částečně, můžeme považovat (Warren, 2013):

- **Hierarchie rodič/potomek**
  V MDM existuje tato speciální hierarchie tvořená tabulkou dimenze a jejím self-joinem, tedy spojením se sama sebou, kdy každý záznam má zároveň odkaz na rodičovský záznam v téže tabulce. Toho je často využíváno při návrhu firemní struktury, kde podřízený je zároveň i nadřízeným. V Tabularu je tato funkcionalita zajištěna funkcí PATH, nicméně na rozdíl od MDM návrhář musí znát maximální hloubku hierarchie již v době návrhu.

- **Relace M:N**
  V multidimenzionálním modelu se jedná o jeden z nejdůležitějších prvků a je využíván velmi často. V Tabularu je sice možné M:N relace taktéž namodelovat, avšak pouze za pomoci poměrně komplexních DAX dotazů. Pokud se takových relací nachází v projektu velké množství, může se stát údržba jeho tabulárního modelu extrémně náročnou.

- **Drilování**
  Díky kterému je možno zobrazit detailní data, která jsou agregována, aby vrátila hodnotu v dané buňce. Drilování je podporováno v obou modelech, přičemž u MDM je možno nadefinovat, které sloupce, dimenze a skupiny měřítek mohou být drilováním vráceny. U Tabularu sice neexistuje žádné takové rozhraní, ale tuto funkcionalitu lze implementovat manuálně za pomoci XMLA definic modelu.

Podrobnější srovnání prvků se nachází také v tabulce na následující stránce (Feldmann, 2013).

Tabulka 6: Srovnání prvků MDM a Tabularu

Prvek	**PowerPivot pro Excel**	**PowerPivot pro SharePoint**	**Analysis Services Tabular**	**Analysis Services Multidimensional**
Množství uživatelů	Jeden, nebo velmi málo (osobní BI)	Malé až střední (týmové BI)	Velké (korporátní BI)	Velké (korporátní BI)
Potřebný software	Office 2010 + PowerPivot	SharePoint 2010 Enterprise, SQL Server 2012 BI nebo Enterprise, PowerPivot pro SharePoint	SQL Server 2012 Enterprise nebo BI	SQL Server 2012 Enterprise nebo BI
Návrhové prostředí	Excel 2010	Excel 2010	SSDT in Visual Studio	SSDT in Visual Studio
Dotazovací jazyk	DAX (MDX je konvertováno na DAX)	DAX (MDX je konvertováno na DAX)	DAX (MDX je konvertováno na DAX)	MDX
Umístění datového modelu	PowerPivot pro Excel	PowerPivot pro SharePoint	SSAS Tabular	SSAS Multidimenzionální
Je možno využít Power View	Ne	Ano (protože využívá DAX)	Ano (protože využívá DAX)	Ne
Typ DB enginu	xVelocity (data komprimována ve vysokém poměru a vložena do paměti)	xVelocity (data komprimována ve vysokém poměru a vložena do paměti)	xVelocity (data komprimována ve vysokém poměru)	OLAP
Velikost datového vzorku	Velikost souboru: 2GB (po kompresi), paměťové omezení: 2GB (32-bit) nebo 4GB (64-bit)	Velikost souboru: 2GB (po kompresi)(omezení nastavené na SharePointu)	Velká (Partitiony, DirectQuery)	Extrémně velká (Partitiony, MOLAP, ROLAP)
Použití různorodých zdrojů dat	Ano, velmi vhodné	Ano, velmi vhodné	Ano, velmi vhodné	Ano, méně vhodné bez vlastního datového skladu či ETL procesů
Možnost dotazovat data přímo na relační úrovni	Ne	Ne	Ano (DirectQuery)	Ano (ROLAP)
Zabezpečení na úrovni řádků	Ne	Ne	Ano (Windows autentizace, filtrování řádků)	Ano (Windows autentizace, dimenzionální/-cellset)
Plánované obnovení dat	Ne	Ano	Ano	Ano
Vývoj ve Visual studiu	Ne	Ne	Ano	Ano
Podpora pro *source control*	Ne	Ne	Ano	Ano
Auditing a management	Ne	Ano (PowerPivot Management Dashboard)	Ano	Ano
Relace M:N	Ano (vytvořené pomocí DAX dotazů, nejsou zabudovány do modelu)	Ano (vytvořené pomocí DAX dotazů, nejsou zabudovány do modelu)	Ano (vytvořené pomocí DAX dotazů, nejsou zabudovány do modelu)	Ano (zabudovány do modelu)
Možnost využití akcí	Drilování (výchozí, nelze přizpůsobit)	Drilování (výchozí, nelze přizpůsobit)	Drilování (Tabular Actions Editor v BIDS Helper)	Drilování, Reporting, Standard

4.7 Výběr správné architektury

Na začátku BI projektu je nutné v SSAS zvolit multidimenzionální nebo tabulární model. Ať už je zvolena jakákoliv možnost, není již později možné u téhož projektu model změnit z jednoho na druhý. Pokud se rozhodneme model změnit, je nutné projekt vybudovat od úplného začátku. Je proto velice důležité důkladně prozkoumat možnosti jednotlivých modelů, aby nedošlo ke ztrátě prostředků a cenného času pro vývoj. Dle hrubých odhadů předních odborníků zabývajících se BISM (Marco Russo a kolektiv) je pro 60 až 70 procent projektů nepříliš podstatné, který model je zvolen a oba budou fungovat stejně dobře. Avšak u zbývajících 30 až 40 procent je podstatné, či dokonce nezbytné, zvolit správný model.

Existuje mnoho kriterií, podle kterých lze rozhodovat o výběru správného modelu. Těmi nejdůležitějšími jsou (Lidberg, 2013):

- **Hardwarové požadavky**
  Multidimenzionální a tabulární model mají velmi rozdílné požadavky na hardware. MDM vyžaduje vysoce výkonné a zároveň vysokokapacitní hard disky. Vyžaduje také RAM paměť, do které se *kešují* data, takže mít dostatek RAM paměti je velmi žádoucí, ale ne však nezbytné jako u Tabularu. Pro Tabular je výkonnost diskového úložiště mnohem nižší prioritou, jelikož využívá in-memory databáze. Proto je u něj mnohem důležitější disponovat s dostatečně velkou RAM pamětí, aby bylo možné držet databázi v paměti i při zvýšeném nároku na její prostředky, například při zpracování dotazů či dat.
  Požadavky na diskové úložiště pro MDM jsou většinou mnohem snadněji splnitelné, než požadavky na RAM paměť u modelu tabulárního, neboť zakoupení velkého objemu diskového prostoru je poměrně levné a obvyklé pro IT oddělení. Mnoho organizací navíc již disponuje SANy[10], které mohou velmi jednoduše poskytnout dostatečný prostor. Naopak zakoupení velkého objemu RAM paměťového prostoru může být velkou překážkou pro jeho finanční náročnost. V současné době stojí 128 GB RAM paměti 2326 dolarů (Kingston, 2014), přičemž u některých rozsáhlejších BI projektů může být RAM paměti zapotřebí několikanásobně více. Může tak nastat situace, kdy je zakoupeno určité množství paměti a časem dojde ke zjištění, že s růstem faktových tabulek, přidáváním dat do modelu a prováděním složitějších dotazů začne docházet k chybám, které pramení z nedostatku paměti. Navíc pro extrémně objemné SSAS implementace s několika terabyty dat může být dokonce nemožné zakoupit dostatek RAM paměti pro uložení modelu, takže je multidimenzionální model jediná možná varianta.

- **Aktualizace předchozí verze SSAS**
  Jak již bylo napsáno, není možné změnit model v průběhu projektu. Proto,

[10]Storage area network je dedikovaná (oddělená od LAN, WAN, atd) datová síť, která slouží pro připojení externích zařízení k serverům (disková pole, páskové knihovny a jiná zálohovací zařízení). (Gupta, 2002)

pokud se již projekt nachází v pokročilé fázi a funguje bez větších nedostatků, pravděpodobně nemá smysl přecházet na tabulární model, který nebyl v předchozí verzi SSAS dostupný. Tento přechod může mít však smysl, pokud se v multidimenzionálně namodelovaném projektu objevují problémy, které by s největší pravděpodobností tabulární model vyřešil.

- **Jednoduchost použití**
  Je pravděpodobné, že pro někoho, kdo nemá zkušenosti s OLAP či multidimenzionálním modelováním, bude Tabular mnohem jednodušší pochopit. Nejen, že jsou jeho koncepty mnohem jednodušší na pochopení – zvlášť, pokud má dotyčný zkušenosti pouze s relačními databázemi –, ale také vývojový cyklus je přímočařejší a obsahuje méně možností. Vybudování prvního tabulárního modelu je zpravidla rychlejší a jednodušší, než vybudování modelu multidimenzionálního.

- **Zpracování dotazů**
  I přes to, že nelze příliš dobře vyvozovat obecné závěry o rychlosti zpracování dotazů, lze konstatovat, že Tabular vykazuje ve většině případů minimálně stejně dobré výsledky jako MDM a v některých specifických případech jeho rychlost překoná. Například měřítka, která kalkulují počty odlišných hodnot (příkaz COUNT DISTINCT), a která jsou obzvlášť slabou stránkou multidimenzionálního modelu, jsou vypočtena extrémně rychle u tabulárního modelu. Zkušenost z již existujících projektů je taková, že dotazy, které mají vrátit v reportu detailní hodnoty (granularita je v tomto případě tedy podobná faktové tabulce), jsou zpracovány u Tabularu mnohem rychleji, za předpokladu, že jsou napsány v jazyce DAX. Pokud však projekt zahrnuje komplexnější výpočty, či modelovací techniky, jako jsou vazby M:N, je mnohem složitější říci, jaký model bude vykazovat lepší výkonnost při zpracování dotazů. (Te Braak, 2013)

- **Rychlost zpracování dat**
  Srovnávat MDM a Tabular z hlediska rychlosti zpracování dat je rovněž obtížné. Zpracování velké tabulky může být v Tabularu mnohem pomalejší, než zpracování ekvivalentní skupiny měřítek v MDM, protože u Tabularu nelze zpracovat partitiony[11] v jedné tabulce paralelně, kdežto u MDM je možno zpracovat partitiony ve stejné skupině měřítek paralelně. Tabular má nicméně tu výhodu, že nepracuje s agregovanými hodnotami, což je jedna z časově náročnějších operací, se kterou se nemusí tedy potýkat. Další výhodou je, že pokud je v Tabularu změněna jedna tabulka, neovlivní to nijak další tabulky. V MDM je naopak nutno počítat s tím, že zpracování dimenze má následné účinky. Zpracování dimenze totiž znamená plné zpracování všech kostek, kde je dimenze použita, což je časově náročná činnost, zvláště pak u rozsáhlejších projektů.

[11] Partitioning je technologie sloužící v relační databázi k fyzickému rozdělení rozsáhlých datových tabulek do menších částí na základě logického členění dat v tabulce, nazývaných partition. (Lobel, 2012)

- **Licencování**
  SQL Server Analyses Services 2012 je dostupný ve třech verzích: Standard, Business Intelligence a Enterprise edition. Ve standardní verzi je však dostupný pouze multidimenzionální model a nabízí stejnou škálu možností, jako jeho předchůdce. To znamená, že zde není možné využít některá podstatná vylepšení MDM, jako je partitioning. Další dvě verze, které jsou finančně nákladnější (BI a Enterprise), umožňují použití MDM a navíc i tabulárního modelu, rozdíl mezi nimi je pouze ve způsobu licencování. Z toho lze dedukovat, že v některých situacích může být využití Tabularu dražší variantou. Pokud tedy rozsah a povaha projektu nevyžaduje partitioning, či využití in-memory databáze a vývojář má zkušenosti s MDM, pak bude pravděpodobně zakoupení verze standard a ušetření financí logickou volbou. Na druhou stranu, pokud tyto požadavky vzniknou a je investováno do BI, či Enterprise verze, neměla by tato volba ovlivnit projektové rozhodování pro jeden, či druhý model.

- **Kompatibilita s programem PowerPivot**
  Tabulární model a PowerPivot jsou v návrhu jejich modelů téměř identické. Uživatelská rozhraní jsou téměř totožná a oba modely využívají jazyka DAX. PowerPivot modely mohou být také importovány do SQL Server Data Tools pro vygenerování Tabulárního modelu. Obráceně ovšem proces nefunguje. Tabulární model tedy nelze převést do PowerPivot modelu.

- **Klientské nástroje**
  Důležitý faktor u BI projektů je také podpora jednotlivých nástrojů pro koncové uživatele. Tabular i MDM podporují MDX dotazy, takže ve většině případů budou klientské nástroje SSAS podporovat oba modely. Lze ze zkušenosti (Singh, 2013) říci, že například nástroje od firmy Microsoft, jako je Excel, či SSRS, fungují stejně dobře u obou modelů, avšak to nemusí platit o nástrojích třetích stran. Některé mohou vyžadovat aktualizaci na nejnovější verzi, nebo nemusí být podporovány vůbec, což by měla za následek jejich nefunkčnost. V současnosti je jazyk DAX podporován pouze v Tabularu, přičemž je do budoucna přislíbena jeho podpora i v MDM. To znamená, že jeden z velmi významných vizualizačních nástrojů – PowerView – je v současnosti podporován pouze v Tabulárním modelu. Ale i poté, co začne multidimenzionální model podporovat DAX, je pravděpodobné, že část funkcionality nástroje PowerView zůstane nedostupná a stejně tak ne veškerá funkcionalita MDM bude dostupná přes DAX dotazy.

- **Real-time BI**
  V posledních letech se požadavek na data obstarávaná v „reálném čase“, čili načtení dat bez delších časových prostojů, stal poměrně běžnou záležitostí. MDM toho dosahuje za použití dvou prostředků. Buď za použití partitionů, nebo využití ROLAP úložiště. První z těchto dvou variant se využívá častěji, přestože může představovat výrazné komplikace při implementaci. ROLAP na druhou

stranu vykazuje nízkou výkonnost při větších objemech dat.
Tabulární model nabízí v podstatě dvě stejné možnosti, ovšem s méně nedostatky jejich MDM ekvivalentu. Pokud jsou data uložena v xVelocity memory enginu, update dat v jedné tabulce nemá žádný vliv na data v ostatních tabulkách, takže zpracování a implementace se stává v konečném důsledku rychlejší. Pokud data zůstavají v relační databázi, pak velkou výhodu představuje ekvivalent ROLAP módu, který se nazývá *DirectQuery*. Tento mód nabízí výrazné zrychlení zpracování dat, neboť překládá veškeré dotazy na SQL a ty pak provádí přímo nad relační DB. Narozdíl od MDM ROLAP, který překládá pouze některé operace do jazyka SQL. Použití DirectQuery však přichází s několika velmi podstatnými omezeními. Za všechny lze vyjmenovat to, že akceptuje pouze DAX dotazy, což znamená, že například uživatelé Excelu nemohou pracovat s real-time daty, protože Excel podporuje pouze jazyk MDX, pouze SQL Server může být použit jako zdroj dat a dále také nejsou podporovány vypočítané sloupce (v SQL Serveru také jako *calculated columns*). (Vercellis, 2013)

5 Analýza databáze Contoso Retail

Následující odstavce jsou věnovány vzorové databázi Contoso Retail, popisují její strukturu a obsah a odpovídají na otázku, proč byla vybrána pro účely této diplomové práce právě tato databáze.

5.1 Historie

K tomu, aby mohli vývojáři, testeři, IT pracovníci i zákazníci otestovat a prezentovat vytvořený softwarový produkt, vznikala často potřeba testovacího vzorku dat, protože reálná data zkrátka nebyla k dispozici. Tento fakt platí i v databázovém odvětví a ani firma Microsoft nezůstává v tomto ohledu pozadu. Již od roku 2005 dává k dispozici ke stažení vzorové databáze, které mají za úkol demonstrovat možnosti využití jejich produktů. Pro tyto demonstrativní účely vytvořila firma Microsoft mimo jiné vzorovou databázi Adventure Works, která představuje data shromážděná fiktivním maloobchodním řetězcem s jízdními koly a doplňky. Tato vzorová OLTP databáze se stala nejrozšířenější a nejpoužívanější v různých prezentacích a demech pro její jednoduchost a kvalitní návrh. S postupem času a rozvojem v oblasti datové analýzy však začala být databáze Adventure Works nedostačující s ohledem na potřebu simulovat komplexní datové struktury a především velké objemy dat ukládané do databází.

5.2 O Contoso datech

V roce 2010 představila firma Microsoft vzorovou databázi Contoso, kterou prezentuje jako *Business Intelligence demo dataset*, který slouží k demonstraci DW/BI funkcionality napříč celou produktovou rodinou Microsoft Office. Tento datový soubor obsahuje fiktivní informace o vedení, prodejích/marketingu, IT a společných finančních scénářích pro maloobchodní prodej. Také podporuje mapovou integraci[12]. Navíc tento dataset nabízí velkoobjemové transfery dat z OLTP a velmi dobře strukturované OLAP databáze společně s referencemi a dimenzionálními daty. (Microsoft, 2014)

Contoso Retail v praxi představuje dva komprimované datové soubory. První z nich je *ContosoBIdemoBAK.exe*, který po dekomprimaci a následném importu poskytuje již vytvořený datový sklad. Druhým souborem je *ContosoBIdemoABF.exe*, což je předpřipravená OLAP kostka, kterou lze použít jako datový zdroj pro analytické a reportingové služby například v programu Excel, SharePoint a dalších. Oba soubory lze najít na webové adrese *http://www.microsoft.com/en-us/download/details.aspx?id=18279*. Po dekomprimaci soubory zabírají zhruba 2 GB místa na disku. Tyto soubory dat jsou tedy oproti ostatním dostatečně rozsáhlé, aby bylo možné demonstrovat nové prvky a možnosti SQL Serveru 2012, jako

[12]Databáze obsahuje souřadnice, které umožňují například analyzovat prodeje dle umístění prodejny

je například partitioning a ColumnStore indexy, ale zároveň dostatečně malé, aby nebylo nutné použít pro jeho zpracování v rozumném čase nákladné/firemní hardwarové prostředky.

Datový sklad Contoso čítá 8 faktových tabulek a 17 dimenzí. Referenční integrita mezi těmito tabulkami je zohledněna pomocí primárních a cizích klíčů. Každá tabulka obsahuje právě jeden primární klíč a může obsahovat i jeden, či několik cizích klíčů. Tabulky také zahrnují sdružený (clustered) index, který slouží ke zrychlení vyhledávacích a dotazovacích procesů v databázi. Kromě toho také datový sklad obsahuje 6 pohledů a 2 uložené procedury. Samotné schéma databáze lze nalézt na konci této práce v kapitole Přílohy.

Velikost a počet řádků jednotlivých tabulek vypadá následovně:

Tabulka 7: Relační databáze Contoso

Tabulka	**Počet řádků**	**Velikost tabulky [MB]**
DimAccount	24	0,008
DimChannel	4	0,008
DimCurrency	28	0,008
DimCustomer	18869	5,852
DimDate	2556	0,867
DimEmployee	293	0,133
DimEntity	421	0,086
DimGeography	674	2,703
DimMachine	7816	4,156
DimOutage	303	0,063
DimProduct	2517	0,961
DimProductCategory	8	0,008
DimProductSubcategory	44	0,008
DimPromotion	28	0,008
DimSalesTerritory	265	0,086
DimScenario	3	0,008
DimStore	310	0,172
FactExchangeRate	773	0,023
FactInventory	8013099	439,050
FactITMachine	23283	0,313
FactITSLA	4925	0,359
FactOnlineSales	12627608	363,552
FactSales	3406089	147,200
FactSalesQuota	7465911	197,800
FactStrategyPlan	2750628	64,750
Celkem	**34326479**	**1228,130**

6 Společný scénář a požadavky obou modelů

Jedním z hlavních cílů každé společnosti je maximalizace zisku. Tento cíl je, mimo jiné, možno realizovat minimalizací nákladů spojených s chodem IT oddělení firmy. Specifičtěji také spojených s údržbou, zpracováním, extrakcí, analýzou a dalšími operacemi s daty. Kde většinou začínají selhávat operativní a nesystematické postupy, získává na významu Business Intelligence se svými nástroji. Kapitola popisuje běžnou situaci, v jaké se firma může nacházet a jakým způsobem může vzniknout potřeba BI řešení.

6.1 Sběr dat

Firma Contoso je maloobchodní firma, která prodává zboží všeho druhu. Elektroniku, spotřebiče, součástky a mnoho dalšího. Řekněme, že tato firma nevyužívá centralizovaný systém. Veškerá data, například o prodejích, zaměstnancích, pobočkách a zákaznících za určité období, jsou ukládána do Excel dokumentů. Tyto dokumenty následně zasílají IT oddělení, které je vkládá do databáze. Také mají k dispozici SharePoint server, kam ukládají vytvořené operativní tabulky a reporty, které sdílejí s ostatními zaměstnanci. Vedení firmy se rozhodlo investovat do nového BI řešení, neboť v případě, že chce provést analýzu na úrovni strategického, či taktického rozhodování, tedy pracovat s daty pokrývající delší časové období a čerpající z různých zdrojů, nemá v současnosti jinou možnost, než požádat o analýzu (či samotná data) IT oddělení. Jejich požadavkem však je, aby data byla již připravená k použití, práce s nimi byla pohodlná, rychlá a personalizovaná, tedy byla různě senzitivní data dostupná různě autorizovaným uživatelům.

6.2 Analýza a interpretace dat

Zaměstnanci a především manažeři jsou velmi dobře seznámeni s nástrojem Microsoft Excel a tvorbou reportů, včetně jeho nástaveb PowerPivot, PowerView a chtěli by tuto znalost využít. Excel umožňuje připojit se k instanci SQL Serveru, a to jak k analytické kostce, tak i samotné databázi. Vytvořit potom za pomocí nástroje PowerPivot (respektive PoverView, či PowerMap) vlastní analýzu je otázkou několika okamžiků. Chtějí si však ponechat možnost delegování této činnosti na IT oddělení pro vytvoření komplexnějších analýz, které vyžadují hlubší znalost BI nástrojů.

6.3 Sdílení

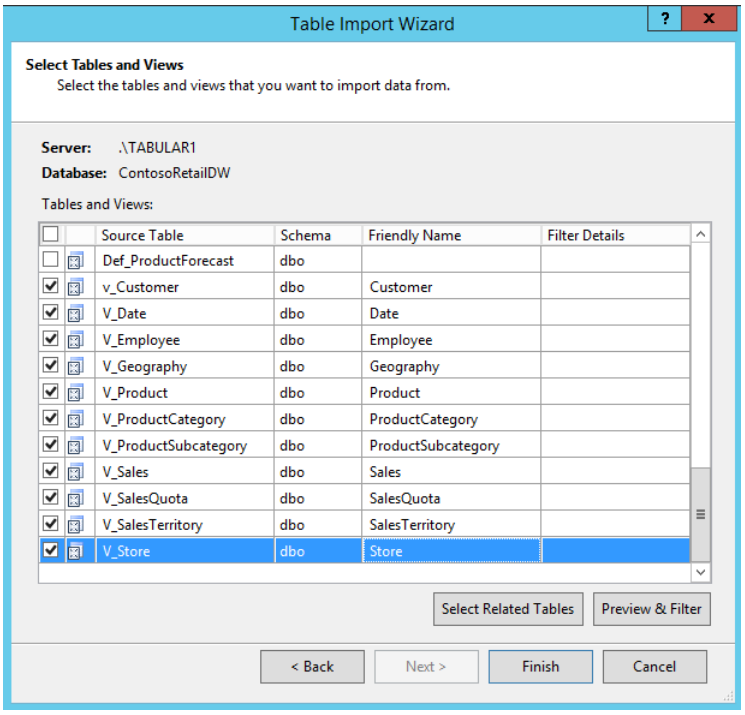
Sdílení již vytvořených reportů si firma přeje zachovat v současné podobě, tedy na SharePoint serveru, jelikož je v tomto ECM systému možné nejen nahrávat vytvořené dokumenty, ale zároveň v nich vyhledávat, tvořit vlastní analýzu, ukládat verzování a spravovat oprávnění jednotlivých uživatelů nebo uživatelských skupin.

7 Datový model

V následujících odstavcích je definován postup firmy Contoso, která za pomocí ETL procesů nashromáždila data a při implementaci Business Intelligence řešení využila možnost tabulární instance SQL Serveru. Obsahuje také srovnání tabulárního modelu s multidimenzionálním, s ohledem na praktickou implementaci.

7.1 Tvorba pohledů

Mezi *best-practices* patří (dle S. Goff, 2013) vytvoření pohledů, které jsou následně importovány do modelu namísto samotných tabulek. S tímto postupem se pojí výhoda úspory času nutného k výpočtům, pokud pohled zahrnuje pouze podmnožinu dat původní tabulky. Tedy například pro Contoso DWH vybere z faktové tabulky pouze několik tisíc řádků a nikoliv celých 10 milionů. Nevýhodou vytvoření pohledů je fakt, že při importu do modelu není automaticky importována také referenční integrita, která se pojí s tabulkami, nikoliv však s pohledy. To znamená, že je nutno vytvořit vazby manuálně. Zde se jedná o práci na úrovni relační databáze, proto se práce z pohledu obou modelů neliší.



Obrázek 5: Import pohledů do modelu (Zdroj: vlastní práce)

7.2 Tvorba modelu

K tomu, aby bylo možné vytvořit datový model, respektive analytickou vrstvu dat a metadat, je nutné vytvořit nový tabulární projekt a připojit se k datovému zdroji. To lze provést v nástroji SQL Server Data Tools (SSDT), který je součástí standardní instalace SQL Serveru 2012. SQL Server nabízí poměrně širokou paletu datových zdrojů, ze kterých lze vybrat. Od relačních (SQL Server, Oracle, Sybase) po NoSQL DB, PowerPivot, či textové soubory. V tomto případě byl zvolen datový sklad. Následně je nutné se připojit ke spuštěné instanci SQL Serveru, specifikovat účet, který přistupuje k datům a následně importovat tabulky, či pohledy. K tomuto importu lze také alternativně využít SQL dotazování namísto využití GUI. Jakmile jsou data importována do modelu, je možné začít pracovat na jeho úpravách a rozšířeních. Multidimenzionální model oproti této variantě využívá nástrojové lišty, kde jsou definovány datové zdroje. Přidání samotného datového zdroje je pak prakticky totožné.

Nástroj SSDT poskytuje dvě základní zobrazení pro práci s modelem. Prvním z nich je tabulkové zobrazení, které vizuálně připomíná Excel tabulku. V tomto zobrazení je obvyklé pracovat v případě, že uživatel chce zobrazit obsah buněk jednotlivých tabulek, vytvořit měřítka, KPI, či odvozené sloupce.

Druhou možností je zobrazení diagramu. Zde je možné pracovat v obecnějším měřítku na úrovni všech tabulek. V zobrazení diagramu je obvyklé pracovat, pokud chceme vytvořit relace mezi tabulkami, přejmenovat je, vytvářet hierarchie, či skrývat před klientskými nástroji jednotlivé sloupce z různých tabulek.

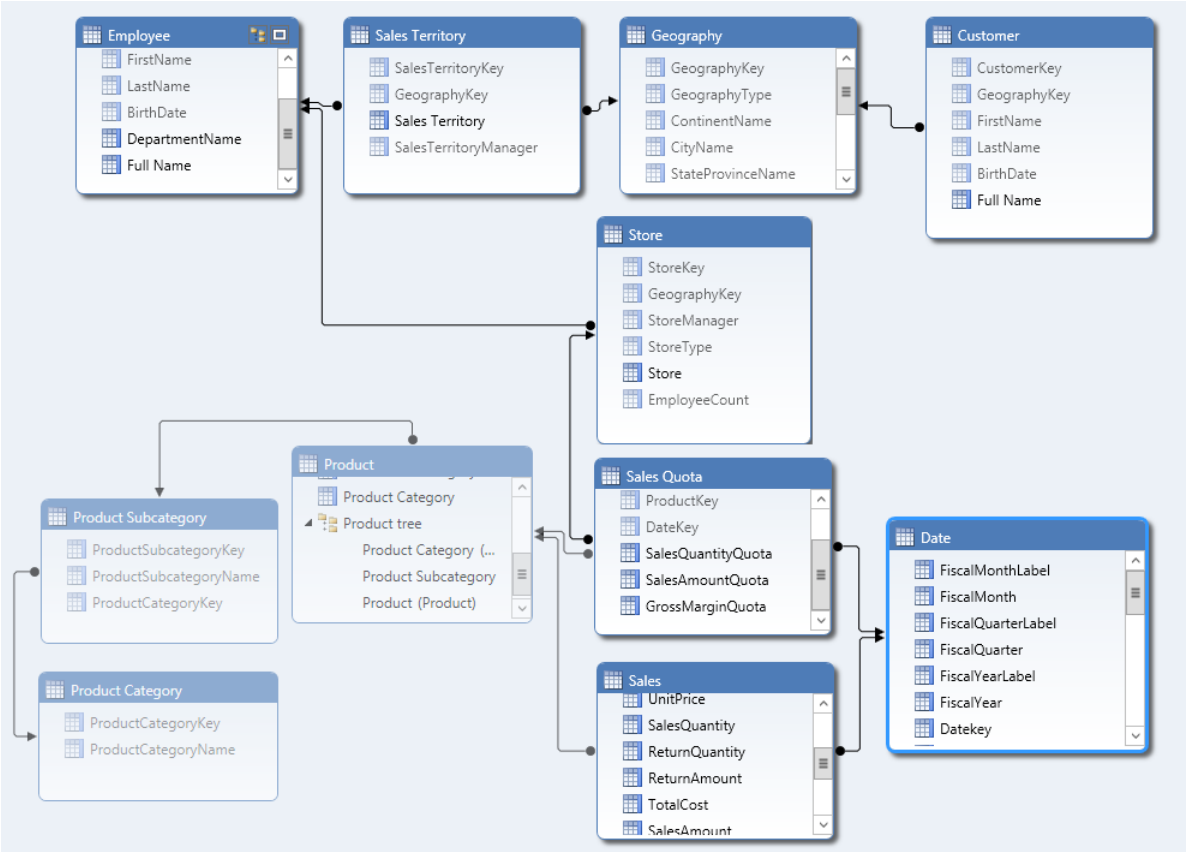
Multidimenzionální model nabízí zcela jiné uživatelské prostředí, kdy se opět za využití nástrojové lišty na pravé straně obrazovky, dělí model do několika částí. Těmi jsou:

- Datové zdroje
- Pohledy datových zdrojů
- Kostky
- Dimenze
- Data-miningové struktury
- Role
- Sestavy

Z tohoto pohledu se tedy může zdát Tabular jako jednodušší varianta díky několika faktorům. Jeho uživatelské prostředí je drtivé většině uživatelů povědomé díky zkušenostem s tabulkovými procesory. Také nabízí menší množství možností, proto může být snazší mu porozumět.

7.3 Tvorba vazeb

Tabulární model má ve tvorbě vazeb několik úskalí. Nepodporuje totiž nativně kardinalitu M:N. Je tedy nutné buď provést normalizaci tabulek předem, nebo ji manuálně vytvořit za pomoci jazyka DAX v modelu. Následně je možné vazby vytvořit přetáhnutím sloupce jedné tabulky, na odpovídající sloupec druhé tabulky. Databáze Contoso byla normalizována tak, aby tento problém nebylo nutno řešit. V multidimenzionálním modelu lze vazby s kardinalitou M:N realizovat bez nutnosti psaní kódu, či normalizace tabulek. Tato funkcionalita je totiž zabudována do modelu nativně.



Obrázek 6: Diagram modelu (Zdroj: vlastní práce)

Dalším úskalím Tabularu může být existence několika vazeb jedné tabulky s druhou tabulkou. Příkladem bývá obvykle faktová tabulka objednávek s několika typy datumů: datum objednání, datum expedice, datum dodání. Všechny tyto datumy referují jeden sloupec v datumové tabulce. Při využití jednotlivých vazeb je následně právě jedna označena jako aktivní, zbytek jako neaktivní. Aktivovat a deaktivovat vazby lze opět manuálně využitím jazyka DAX, kde jsou vytvořena měřítka pro jednotlivé datumy a za pomoci fuknce USERELATIONSHIP upřesněno, ke kterému datu se měřítko vztahuje.

Zdrojový kód 1: Definice aktivní vazby

```
1 SalesDueDate:=CALCULATE(sum(Sales),USERELATIONSHIP('Date'[↩
    DateKey], Sales[DueDateKey]))
```

Každé měřítko je tak nutné manuálně vytvořit tolikrát, kolik vazeb z jedné tabulky do jednoho sloupce druhé tabulky existuje. V případě popsaném výše by se tedy počet měřítek ztrojnásobil.

Multidimenzionální model nabízí oproti Tabulárnímu elegantnější řešení. Vytvoří automaticky pro každou vazbu pohled (view), který vystupuje jako samostatná dimenze. Je potřeba se tedy pouze přesvědčit, že v kostce daná vazba existuje. To lze udělat otevřením příslušné kostky a v nabídce *Dimension Usage* přiřadit příslušné dimenze k měřítkům.

7.4 Tvorba měřítek

Dalším krokem může být vytváření měřítek. Ty se v tabulárním režimu vytvářejí pomocí GUI a jazyka DAX v tabulkovém zobrazení. Jedna z prvních věcí, co bude firmu zajímat, jsou jednoznačně analýza výnosů a nákladů. Prvními vytvořenými měřítky budou obrat a náklady. Toho lze docílit označením příslušného sloupce a zvolením sumy (řecké písmeno sigma) v liště panelů. V prostoru pod dělicí čárou je vložen například následující kód:

Zdrojový kód 2: Suma tržeb

```
1 Sales:=SUM([SalesAmount])
```

V multidimenzionálním modelu lze vytvořit měřítka a odvozené sloupce opět na úrovni kostky (více na http://msdn.microsoft.com/en-us/library/ms365347.aspx). Kromě odlišného grafického rozhraní a postupu jejich tvorby se funkcionalita od tabulárního modelu nijak neliší. Následně vypočteme totéž i pro náklady. Po odečtení dostaneme hrubý zisk a jeho procentuální vyjádření. Jednou z velmi jednoduchých a užitečných funkcí Tabularu je funkce *TOTALYTD*. Za pomocí této funkce lze velmi rychle a pohodlně spočítat kumulativní částku za období jednoho roku.

Zdrojový kód 3: Kumulativní funkce TOTALYTD

```
1 Sales YTD:=TOTALYTD(Sales[Sales];'Date'[Datekey];All('date'))
```

Na obrázku 7 je vidět, že zatímco v prvním sloupci označeném modře je zisk v jednotlivých měsících, v druhém (zeleném) sloupci se nachází kumulativní zisk, tedy součet z aktuálního období přičtený k zisku ze všech předchozích měsíců aktuálního roku. V třetím, červeně označeném sloupci, je pak pro srovnání kumulativní zisk z minulého roku. K tomu, aby tato funkce měla správnou funkcionalitu, je nutné označit jednu z tabulek modelu jako „datumovou“ a zvolit v této tabulce idetifikátor datumu ve formě *datetime*.

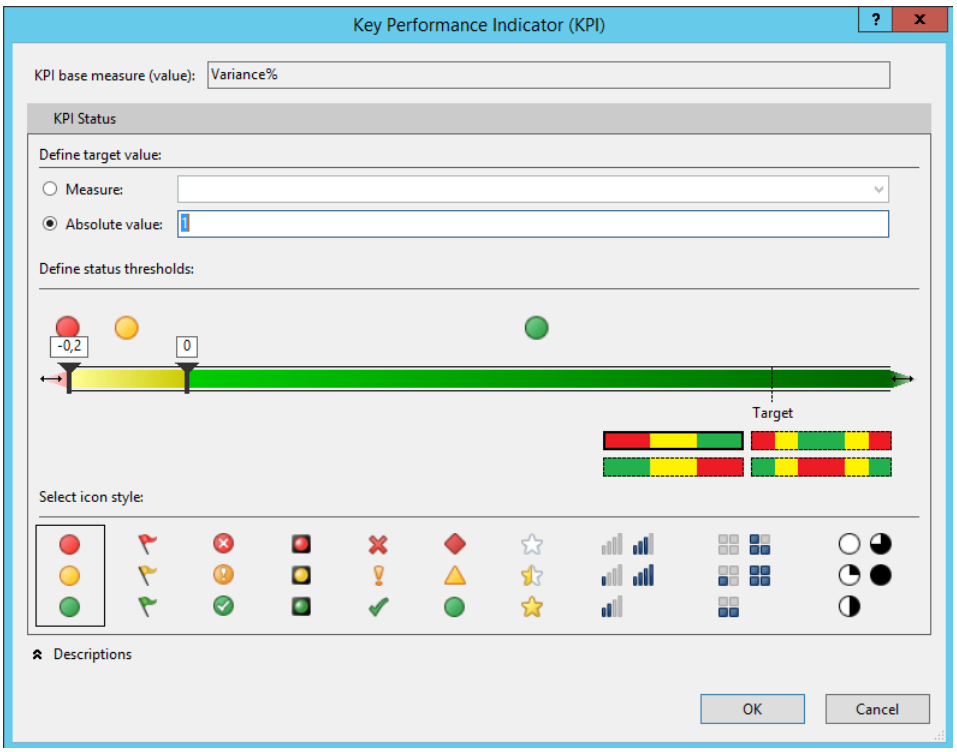
Row Labels	Sales	Sales YTD	Sales PYTD	Variance	Variance%	Variance% Status
2008	**5 962 909,36 Kč**	**5 962 909,36 Kč**	**6 519 005,88 Kč**	**-1 519 135,88 Kč**	**-20,30 %**	
January	456 327,97 Kč	456 327,97 Kč	454 930,41 Kč	5 551,85 Kč	1,23 %	
February	480 538,81 Kč	936 866,78 Kč	920 472,48 Kč	-56 856,80 Kč	-10,58 %	
March	442 566,70 Kč	1 379 433,48 Kč	1 272 275,79 Kč	-33 017,48 Kč	-6,94 %	
April	426 930,82 Kč	1 806 364,31 Kč	2 014 379,48 Kč	-355 723,87 Kč	-45,45 %	
May	724 316,24 Kč	2 530 680,54 Kč	2 763 933,74 Kč	-78 504,50 Kč	-9,78 %	
June	359 679,69 Kč	2 890 360,23 Kč	3 297 688,39 Kč	-182 122,43 Kč	-33,61 %	
July	551 261,14 Kč	3 441 621,37 Kč	3 851 881,85 Kč	-269 890,17 Kč	-32,87 %	
August	304 507,30 Kč	3 746 128,68 Kč	4 313 307,52 Kč	-152 871,34 Kč	-33,42 %	
September	421 194,55 Kč	4 167 323,22 Kč	4 808 832,35 Kč	-291 606,58 Kč	-40,91 %	
October	605 669,49 Kč	4 772 992,71 Kč	5 394 450,55 Kč	779,00 Kč	0,13 %	
November	601 219,11 Kč	5 374 211,82 Kč	5 890 568,00 Kč	57 059,58 Kč	10,49 %	
December	588 697,54 Kč	5 962 909,36 Kč	6 519 005,88 Kč	-161 933,16 Kč	-21,57 %	
Grand Total	5 962 909,36 Kč	5 962 909,36 Kč	6 519 005,88 Kč	-1 519 135,88 Kč	-20,30 %	

Obrázek 7: PowerPivot (Zdroj: vlastní práce)

Funkci YTD lze použít i u multidimenzionálního modelu v jazyce MDX, avšak její použití je o něco složitější, což vyplývá z vyšší komplexnosti základních formulí jazyka MDX.

7.5 KPI

KPI (klíčové ukazatele výkonnosti) jsou vytvořeny označením měřítka a volbou *Vytvořit KPI*. Zde lze najít, narozdíl od multidimenzionálního modelu, poměrně komfortní grafickou nástavbu. Není tedy nutné využít příkaz CASE a definovat podmínky jednotlivě. Zde jsou na výběr dvě možnosti výsledné hodnoty. Buď je možné zvolit vypočtenou hodnotu z výrazu, nebo absolutní hodnotu. Dále jsou k dispozici na výběr různá grafická vyjádření klíčových indikátorů ve spodní části okna. Výsledné KPI pak sestává z položek Value, Goal, Status a Trend, které lze přidat do reportu, viz obr. 7, sloupec *Variance% Status*.



Obrázek 8: Tvorba klíčových ukazatelů výkonnosti (Zdroj: vlastní práce)

7.6 Tabulka s datumy

Mezi tzv. *best-practices*, tedy osvědčené postupy v BI development prostředí, také patří vytvoření samostatné tabulky, která slouží pro určení datumu (ale také fiskálního období, pololetí, čtvrtletí, měsíce, dne v týdnu a podobně). V takové tabulce by měl být jeden záznam pro každý den (v některých specifických případech i více), aby nevznikaly mezery a problémy s kvalitou dat. V Contoso databázi se taková tabulka již sice nachází, ale pokud je nutné ji zde z nějakého důvodu doplnit do modelu (například v případě, že je model vytvářen z excel dokumentů a datumová data nejsou k dispozici), je možno využít některou předdefinovanou datumovou tabulku, například ze služby *ODATA feed*, nebo ze stránky CodeProject (CodeProject, 2014). Tuto tabulku je vhodné, jak již bylo řečeno, označit jako datumovou v nabídce Table → Date → Mark As Date table a doplnit její vazby na další tabulky, případně ji upravit tak, aby korespondovala s požadavky na model.

7.7 Partitioning

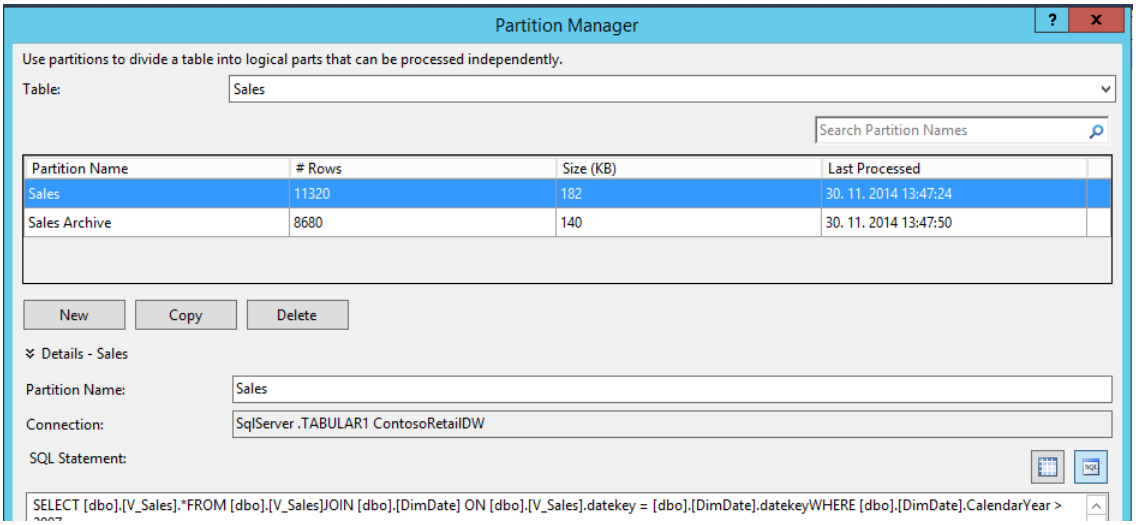
Tabulární režim nabízí, stejně jako ten multidimenzionální, rozdělení tabulky na logické části, nebo-li partitioning. Toho lze dosáhnout zvolením volby Table → Partitions. Zde je možné rozdělit tabulku za pomoci SQL dotazu. SQL dotaz, který vytvoří partition pro rok 2008 bude vypadat následovně:

Zdrojový kód 4: SQL dotaz pro vytvoření partitionu

```
1 SELECT * FROM Sales s~JOIN Date d ON s.datekey = d.datekey ↩
    WHERE d.year = 2008
```

Tabular zde nabízí poměrně intuitivní a jednoduché rozhraní. Na rozdíl od multidimenzionálního modelu však SQL Server 2012 nezpracovává jednotlivé partitiony současně, ale jeden po druhém. Navíc neumožňuje uložit preagregované hodnoty. Obvykle se partitioning provádí, pokud je množství položek větší, než zhruba 20 až 25 miliónů záznamů, přičemž faktová tabulka s prodeji obsahuje něco pod 10 miliónů řádků. Pro demonstraci však byla rozdělena na partitiony dle jednotlivých let na položky z aktuálního roku, posledních dvou let a starších. Tím je umožněno frekventovaněji aktualizovat určité množství dat, například ta nedávno nashromážděná. Déle uchovávaná data, která se tolik nemění, jsou aktualizována naopak méně často. To má za následek snížení zátěže kladených na systémové prostředky serveru.

Partitiny v multidimenzionálním modelu lze spravovat v kostce na kartě Partitions taktéž za pomoci SQL dotazu. Co se týče implementace partitionů, oba modely se jeví jako rovnocenné. To stejné se ovšem nedá říct o zpracování partitionů, ve kterém má, díky výše zmíněným vlastnostem, multidimenzionální model navrch.



Obrázek 9: Tvorba partitionů (Zdroj: vlastní práce)

7.8 Perspektivy

Perspektivy umožňují uživatelům zmenšit množinu dat, se kterou pracují při provádění analýzy. Takže namísto všech vidí pouze relevatní data pro jednotlivou osobu či například oddělení. V Tabularu se vytváří pomocí nabídky Model → Perspektives → Create and Manage. Následně je vytvořena nová perspektiva a vybráno, jaké tabulky a jejich sloupce budou uživateli zobrazeny. V tabularu, jakožto ani v MDM, nelze tuto funkcionalitu použít jako zabezpečení. Uživatel si totiž může vybrat perspektivu sám. V multidimenzionálním modelu je perspektiva definovaná na kartě *Perspectives* na úrovni kostky. Oba modely nabízí v tomto ohledu totožnou funkcionalitu.

7.9 Zabezpečení/Role

Zabezpečení je možno zajistit za pomoci nastavení rolí, kterým jsou přiřazena práva číst data na úrovni instance serveru, databáze nebo řádku. Naopak autorizaci nelze v tabularu zabezpečit na úrovni tabulky (pouze částečně), sloupce, buňky, či perspektivy.

7.9.1 Přisvojení osobnosti

Impersonation, tedy přisvojení profilu jiného uživatele, slouží k autentizaci a následné autorizaci provádění operací s daty. Tato operace by se dala dále rozdělit mezi serverovou a klientskou. Mezi osvědčené postupy na straně serveru patří využívat servisní účet, který má nastavená vysokoúrovňová práva, aby byl schopen provádět například import a zpracování dat. Na straně klienta by pak mělo do-

jít k autentizace pomocí osobního účtu, který má obvykle nižší autorizační práva a slouží k prohlížení a filtrování dat, práci s tabulkami, správu partitionů a podobně.

7.9.2 Zabezpečení na úrovni serveru

Pravým kliknutím na tabulární instanci v SSMS a přepnutím na kartu Security je možné nastavit uživatelská práva na úrovni serveru. Přidáním uživatele do této skupiny získá uživatel administrátorská práva k veškerému obsahu, který se na této instanci serveru nachází. Proto je potřeba důkladně zvážit, kterému uživateli tato práva poskytnout. Také je vhodné zvážit, zda mají administrátoři stroje být zároveň také administrátory serveru. Tuto možnost je možné deaktivovat opět v nabídce Security → General → Advanced Features → BuiltInAdminsAreServerAdmins.

7.9.3 Zabezpečení na úrovni řádku

Řádkové filtry jsou definovány za pomoci jazyka DAX v nabídce Model → Roles. Filtry, které umožňují roli "North America Computers"zobrazit pouze počítačové produkty ze Severní Ameriky, vypadají následovně:

Zdrojový kód 5: Řádkový filtr

```
1 Geography[ContinentName] = "North America"
2 Product[Category] = "Computers"
```

Zde může nastat problém, když máme normalizované tabulky například pro produkt, jeho kategorie a podkategorie. Pokud je filtr aplikován pouze na jednu z těchto tabulek, může to vést ke zmatení a nespokojenosti uživatele, jelikož jsou zobrazeny „nedostupné“ kategorie, které přestože jsou zobrazeny ve filtrech, neobsahují žádné produkty. Řešením je zpětná denormalizace v tabulce produkt, kdy je ke každému produktu přiřazena jeho kategorie a podkategorie, viz zdrojový kód č. 15.

Zdrojový kód 6: Denormalizace

```
1 RELATED('Product Category'[ProductCategoryName])
```

Dále je vhodné skrýt tyto přiřazené tabulky před klientskými nástroji pomocí pravého tlačítka a nabídky „Hide from client tools“. Další problém může nastat, pokud je nutno implementovat velmi specifická kritéria filtrování. Jinými slovy, je nutno aplikovat mnoho různých podmínek. Pokud jsou filtry aplikovány, jako jedna podmínka za druhou, pomocí DAX kódu, stane se kód velmi špatně udržovatelným a nepřehledným. Zde se nabízí použití dynamického zabezpečení, které spočívá ve vytvoření tabulky, ve které jsou uloženy unikátní kombinace filtrů spojené s rolemi v této tabulce.

Tabulka 8: Tabulka zabezpečení

User Name	**Allowed sales territories**
FOREST\Administrator	198
FOREST\Administrator	200

V definici role je pak vložena DAX formule (Collie, 2012), která za pomoci vyhledávací funkce LOOKUPVALUE přidělí příslušné autorizace pro uživatele v tabulce Security. Funkce USERNAME() pak vrací řetězec ve formě *DOMÉNA\jméno.*

Zdrojový kód 7: Dynamická funkce řádkového filtru

```
1 ='Sales Territory'[SalesTerritoryKey]=
2   LOOKUPVALUE( Security[Allowed sales territories],
3                Security[User Name], USERNAME(),
4                Security[Allowed sales territories],
5                'Sales Territory'[SalesTerritoryKey]
6              )
```

Následně jsou v tomto scénáři pro uživatele přihlášeného jako *FOREST\Administrator* dostupná pouze teritoria s primárním klíčem 198 a 200.

V multidimenzionálním modelu jsou definovány role za pomoci grafického rozhraní a není nutné napsat kód k zabezpečení této funkcionality. Ten může být generován i pomocí wizardu na záložce *Dimension Data*. Multidimenzionální model kromě toho ale nabízí jeden prvek, který v Tabulárním modelu chybí. Jedná se o tzv. „Non Visual Totals“. Tento checkbox lze nalézt při definici role na téže záložce. Pokud tento checkbox ponecháme nezaškrtnutý, tak jsou uživateli zobrazeny nejen agregované hodnoty z dat, která jsou pro uživatele viditelná, ale i z těch, která viditelná nejsou. Pokud tedy uživatel má přístup pouze například ke dvěma trhům z celkových desíti, suma prodejů bude čítat prodeje ze všech deseti trhů. Tento prvek není příliš využívaný, ale může se objevit scénář, ve kterém firma právě takové chování modelu požaduje. V tom případě, alespoň tedy ve verzi SQL Serveru 2012, musí zvolit multidimenzionální model.

8 Potenciál tabulárního modelu v praxi

Síla tabulárního modelu tkví především v nižší komplexnosti, kterou představuje jeho životní cyklus oproti modelu multidimenzionálnímu. Jinými slovy, takový model vyžaduje nižší nároky na čas potřebný k jeho vývoji a údržbě, což bylo prokázáno i v předešlých kapitolách. Tato přednost však není to jediné, co může tabulární model nabídnout. Někdy se může tabulární model i u rozsáhlých projektů stát vítěznou variantou nad klasickým multidimenzionálním modelem. V této kapitole je nastíněn takový scénář, kdy mladší Tabular překonává svého staršího sourozence.

8.1 Distinct count

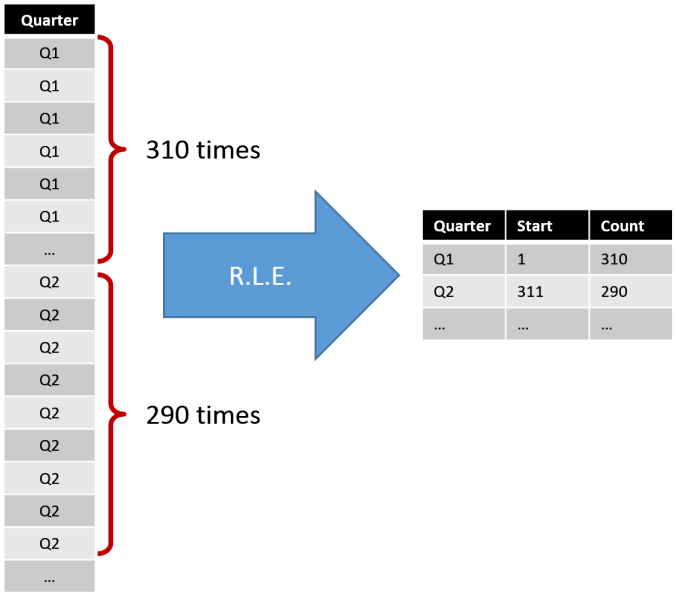
Jedním z nejdůležitějších požadavků na model může ve specifických případech být rychlá kalkulace rozdílných záznamů – v řeči SQL – DISTINCT COUNT. Jedná se o jednu z nejvíce využívaných analýz, zároveň však jeden z nejtěžších problémů, které se v OLAP řeší. Typickým příkladem takové situace může být kalkulace v oblasti prodeje a marketingu. Mějme kostku, která analyzuje prodeje. Mezi dimenze kostky patří geografické údaje o zákazníkovi, vzdělání, mzdové rozpětí a pohlaví. Také popisuje kategorii produktu, model, velikost, čas a obchodního zástupce. Měřítka zahrnují informace o prodejích, kvantitu a slevu.

Jedním z nejčastějších dotazů zní: „Kolik zákazníků nakoupilo specifický produkt v určitém časovém období?“ Obecnější otázka pak zní: „Kolik zákazníků kupuje jednotlivé produkty?“

Ačkoliv poslední otázka zní jednoduše, opak je pravdou. Pokud je použita agregační funkce COUNT, pak neposkytuje správné výsledky, neboť se ve výsledku mohou vyskytnout duplikáty. Více o tomto problému lze najít na http://technet.microsoft.com/en-us/library/aa902680(v=sql.80).aspx.

Tato slabina multidimenzionálního modelu plyne z faktu, že jsou záznamy uloženy v databázi po řádcích. V případě, že je proveden příkaz COUNT DISTINCT, musí být prohledány všechny řádky v tabulce a jejich hodnoty porovnávány s doposud vybudovaným slovníkem. Tato operace může být velice časově náročná, pokud je prováděna na velkém objemu dat.

Tabular oproti tomu data ukládá do jednotlivých sloupců. Každý sloupec je komprimován. Komprese Tabularu (jak uvádí A. Ferrari, 2014) je sice proprietární, ale velmi se podobá kompresi RLE. Taková komprese namísto mnohonásobného ukládání též hodnoty uloží tuto hodnotu pouze jednou a následně uloží počet jejich opakování.



Obrázek 10: Run Length Encoding (Zdroj: A. Ferrari, 2014)

xVelocity engine pak vytvoří slovník, jak je popsáno v kapitole 4. Tento fakt je podstatný, neboť operace DISTINCT COUNT pak může být provedena s výhodou již vybudovaného slovníku, což může podstatně zrychlit dotazování a v tomto případě v globálním měřítku rozhodnout ve prospěch tabulárního modelu.

8.2 Optimalizace modelu

Rychlost výpočtů nad tabulárním modelem je závislá na více faktorech. Jedním z nejdůležitějších je počet rozdílných hodnot ve sloupci. Vezměme například v úvahu sloupec čtvrtletí. Ten obsahuje pouze čtyři hodnoty pro každý záznam tabulky. Kompresní algoritmus tak bude velmi efektivní, neboť bude schopen uložit hodnoty do minimální možné velikosti, což jsou 2 bity.

Oproti tomu jiný sloupec, pro ilustraci délka trvání telefonního hovoru, může představovat pro kompresní algoritmus problém. Pokud je čas měřen na milisekundy, případně setiny sekundy, každý hovor má jinou délku. Z analytického hlediska je možné usoudit, že je nepodstatné uchovávat čas při takto vysoké granularitě. Snížením granularity například na desetiny sekundy je možné razantně snížit počet rozdílných hodnot, což má za důsledek značné vylepšení výkonu dotazů a snížení nároků na úložný prostor RAM paměti.

8.3 Příprava dat pro testování

Vzhledem k tomu, že tabulární data jsou kompletně načtena do RAM paměti, dá se předpokládat, že rychlost dotazování a práce s těmito daty bude pro menší až středně velký vzorek dat (desítky GB) i bez optimalizace modelu velice rychlá. Analytická

databáze Contoso po vytvoření modelu a nasazení na server zabírá v RAM paměti (a na disku) 379 MB. Oproti relační databázi, která zabírá na disku 1.22 GB tak reálně došlo k více než trojnásobné kompresi. Kompresní poměr tabulárního modelu (dle Russo, 2013) však může dosáhnout poměru až 1:100 především v závislosti na struktuře dat. Schéma modelu lze vidět výše na obrázku 6.

Po vytvoření modelu a nasazení tabulární analytické databáze lze využít PowerPivot tabulku (De Jonge, 2012), která slouží k analýze její alokované RAM paměti.

	Object Memory Usage MB	Pct of total
WIN-B0K2KS6TU0J\TABULAR1	1 682,92	79,9 %
ContosoModel	265,27	12,6 %
(blank)	0,07	0,0 %
CalculatedColumns	0,00	0,0 %
Cubes	4,46	0,2 %
Data Source Views	0,06	0,0 %
Data Sources	0,02	0,0 %
DBAssemblies	0,00	0,0 %
Dimensions	260,64	12,4 %
Customer	1,21	0,1 %
Date	1,15	0,1 %
Employee	0,35	0,0 %
Geography	0,35	0,0 %
Measures	3,09	0,1 %
Product	0,64	0,0 %
Product Category	0,15	0,0 %
Product Subcategory	0,17	0,0 %
Sales	52,70	2,5 %
Sales Quota	200,32	9,5 %
Sales Territory	0,18	0,0 %
Store	0,32	0,0 %
Mining Structures	0,00	0,0 %
Permissions	0,01	0,0 %
Roles	0,01	0,0 %

Obrázek 11: Využití RAM paměti jednotlivými položkami analytické databáze

Některé menší podniky se objemem dat budou podobat tomuto modelu. Pokud tyto podniky nebudou omezovat limitované možnosti tabulárního modelu, je pravděpodobné, že se jim použití tohoto modelu vyplatí, jelikož se jedná o přímočařejší, rychlejší a v konečném důsledku i úspornější vývoj BI řešení. Tabular, přestože se jedná spíše o výjimečné případy, může být výhodný i tehdy, když se objem dat zpracovaný firmou pohybuje v řádech několika stovek gigabajtů až několika terabajtů. V takovém případě je již naprosto nezbytné se zabývat optimalizací modelu, neboť uspořené serverové prostředky mohou znamenat značnou redukci nákladů na BI řešení, zvýšení rychlosti práce s daty, či snížené vytížení serveru. Bylo proto nezbytné pro účely této práce rozšířit Contoso databázi na několikanásobek původní velikosti, aby její výsledky poskytly dostatečnou důvěryhodnost a rozdíly v použití mezi jednotlivými instancemi byly dostatečně patrné.

8.4 Testovací server

K tomu, aby bylo možné provést testování, které poskytne důvěryhodné výsledky, bylo potřeba nakládat s dostatečně velkým objemem dat. Jak již bylo v textu zmíněno, tabulární model ukládá veškerá data do RAM paměti. Náročnost na hardwarové prostředky tak prakticky vyloučila možnost použití domácího stroje, neboť by veškeré dotazování bylo možné provést velmi rychle (právě díky nízkému objemu dat limitovaného velikostí RAM paměti). V současnosti je však trend takový, že firmy ukládají velké objemy dat (v řádech několika stovek gigabajtů až několika terabajtů) a ve většině firem se objem dat, které je nutno uchovat a zpracovat, s každým rokem značně zvyšuje. Proto byl pro potřeby této práce dočasně zřízen virtuální server s následujícími parametry:

Tabulka 9: Vybavení virtuálního serveru

Parametr	**Hodnota**
CPU	Intel Xeon X5660 2.8 GHz
RAM	64 GB DDR3
HDD	540 GB SATA
Operační systém	Windows Server 2012 R2
Databázový systém	SQL Server 2012 Enterprise

Po využití scriptu (Příloha A) narostl objem nové faktové tabulky pojmenované FactSalesBig na 353 násobný objem oproti původní tabulce FactSales. Detaily obou tabulek lze nalézt v následující tabulce:

Tabulka 10: Zvětšení tabulky FactSales

	Tabulka FactSales	**Tabulka FactSalesBig**
Počet řádků	3 406 089	1 015 014 522
Velikost v datovém skladu [MB]	147,2	52169,76
Velikost v tabulární analytické DB [MB]	123	14688,38

Nyní vypadá alokace RAM paměti analytické tabulární databáze následovně:

⊟Dimensions	14 667,78	57,5 %
⊞(blank)	0,01	0,0 %
⊞DimCustomer	4,03	0,0 %
⊞DimDate	1,52	0,0 %
⊞DimEmployee	1,30	0,0 %
⊞DimGeography	0,54	0,0 %
⊞DimProduct	1,87	0,0 %
⊞DimProductCategory	0,29	0,0 %
⊞DimProductSubcategory	0,46	0,0 %
⊞DimStore	1,19	0,0 %
⊟FactSalesBig	14 653,99	57,5 %
⊞(blank)	0,26	0,0 %
⊞(All)	0,00	0,0 %
⊞DateKey	0,00	0,0 %
⊟In-Memory Table	14 653,68	57,5 %
⊞(blank)	0,00	0,0 %
⊟Columns	10 781,50	42,3 %
⊞(blank)	0,00	0,0 %
⊞DateKey	1 548,89	6,1 %
⊞ProductKey	1 548,88	6,1 %
⊞ReturnAmount	272,49	1,1 %
⊞ReturnQuantity	0,03	0,0 %
⊞RowNumber	0,02	0,0 %
⊞SalesAmount	606,93	2,4 %
⊞SalesKey	3 871,99	15,2 %
⊞SalesQuantity	0,08	0,0 %
⊞StoreKey	1 106,31	4,3 %
⊞TotalCost	928,84	3,6 %
⊞UnitCost	544,79	2,1 %
⊞UnitPrice	352,26	1,4 %
⊞Hierarchies	3 872,18	15,2 %

Obrázek 12: Využití RAM paměti po zvětšení tabulky FactSales

8.5 Testovací scénáře

Po rozšíření datového vzorku bylo nutné vytvořit testovací scénáře, ve kterých by bylo možno otestovat rychlost zpracování analytických dat. Jedním z nejběžnějších požadavků organizace je analýza finančních ukazatelů, jako jsou příjmy, výdaje, tržby, zisk, případně porovnání těchto veličin v čase. Dalším požadavkem zpravidla bývá analýza nákupního chování zákazníka, nákupu jednotlivých produktů v závislosti na datu, geografii a podobně.

8.5.1 Scénář 1 – Rozpad produktů na jednotlivé dny

Firma Contoso požaduje po svém BI oddělení informaci, které kategorie produktů jsou prodávány v jednotlivých dnech dvoutýdenního předvánočního období (10-23. prosince).

Tabulární model
Jazyk DAX umožňuje pracovat s analytickými daty podobně, jako jazyk SQL s těmi

relačními. EVALUATE slouží jako obdoba příkazu SELECT, jedná se tedy o nezbytný příkaz pro práci s daty. Dále příkaz FILTER, který podobně jako příkaz WHERE, slouží k výběru podmnožiny dat dle určitého kritéria. Nakonec příkaz SUMMARIZE, který slouží jak ke spojování (JOIN) tabulek, tak k seskupování dat (GROUP BY v SQL).

Zdrojový kód 8: Prodeje produktů v předvánočním období (DAX)

```
1  evaluate
2  (
3    filter
4    (
5      summarize
6      (
7        Sales
8        , Sales[DateKey]
9        , "Sum of Sales"
10       , Sum(Sales[SalesAmount])
11       , "Distinct Products"
12       , DistinctCount(Sales[ProductKey])
13     )
14     , (Sales[DateKey]<"24. 12. 2007 0:00:00")
15   )
16 )
17 order by
18 Sales[DateKey]
19 Start at
20 "1. 12. 2007 0:00:00"
```

Multidimenzionální model

Jazykový konstrukt MDX nabízí standardní analýzu multidimenzionálních dat za pomoci definování informací viditelných v jednotlivých řádcích a sloupcích. Zde je filtrování provedeno pomocí vnořeného SELECT dotazu, který vybere potomky dimenze date odpovídající zadanému kritériu. Filtrování prázdných hodnot je provedeno pomocí příkazu NON EMPTY. Použitý MDX dotaz, který koresponduje z dotazem DAX, je vždy možno nalézt v sekci Přílohy (Příloha C) na konci práce.

Výkon jednotlivých modelů lze nalézt v následující tabulce:

Tabulka 11: Výkon jednotlivých modelů – scénář 1

Pokus	**Čas zpracování dotazu [s] (DAX)**	**Čas zpracování dotazu [s] (MDX)**
1	48	95
2	48	94
3	46	98
4	47	102
5	48	94
∅	**47,4**	**96,6**

8.5.2 Scénář 2 – Počet zákazníků v jednotlivých městech

Vedení firmy Contoso se chce dozvědět více informací o tom, ze kterých pochází největší množství zákazníků. Zajímají ji ta města, kde počet unikátních zákazníků v jednotlivých městech přesáhne určitou hranici (v tomto případě 100 zákazníků).

Tabulární model

Dotaz, který byl použitý v tomto scénáři operuje navíc s vytvořeným měřítkem *Customers*, které agreguje unikátní zákazníky.

Zdrojový kód 9: Počet zákazníků v jednotlivých městech (DAX)

```
1  evaluate
2  (
3    filter
4    (
5      summarize
6      (
7        Geography
8        , Geography[CityName]
9        , "Unique customers"
10       , [Customers]
11
12     )
13     , [Customers] > 100
14   )
15 )
16 order by
17 [Customers] DESC
```

Rychlost zpracování uvádí tato tabulka:

Tabulka 12: Výkon jednotlivých modelů – scénář 2

Pokus	**Čas zpracování dotazu [ms] (DAX)**	**Čas zpracování dotazu [ms] (MDX)**
1	33	38
2	32	35
3	37	32
4	33	37
5	34	33
∅	**33,8**	**35**

8.5.3 Scénář 3 – Objem prodejů jednotlivých manažerů poboček

Nakonec by se vedení rádo dozvědělo, jak si vedli jednotliví manažeři prodejen s ohledem na prodeje.
Kromě výše popsaných funkcí je zde uvedena funkce SUMX, která umožňuje agregovat data pro každý řádek určité tabulky. Jedná se tedy v podstatě o konkrétní určení kontextového filtru, které funkce SUM neumožňuje.

Zdrojový kód 10: Objem prodejů jednotlivých manažerů poboček (DAX)

```
1  evaluate
2  (
3    filter
4    (
5      summarize
6      (
7        Employee
8        , Employee[Full Name]
9        , "Sum of Sales"
10       , Sum(Sales[SalesAmount])
11       , "Sales %"
12       , Sum(Sales[SalesAmount])/Sumx(all(Sales), Sales[↩
             SalesAmount])*100
13     )
14     , Sum(Sales[SalesAmount]))
15   )
16 order by
17 Employee[Full Name]
```

Výkon jednotlivých modelů lze nalézt v následující tabulce:

Tabulka 13: Výkon jednotlivých modelů – scénář 3

Pokus	**Čas zpracování dotazu** [s] **(DAX)**	**Čas zpracování dotazu** [s] **(MDX)**
1	9	8
2	7	8
3	7	7
4	7	7
5	6	7
∅	**7,2**	**7,4**

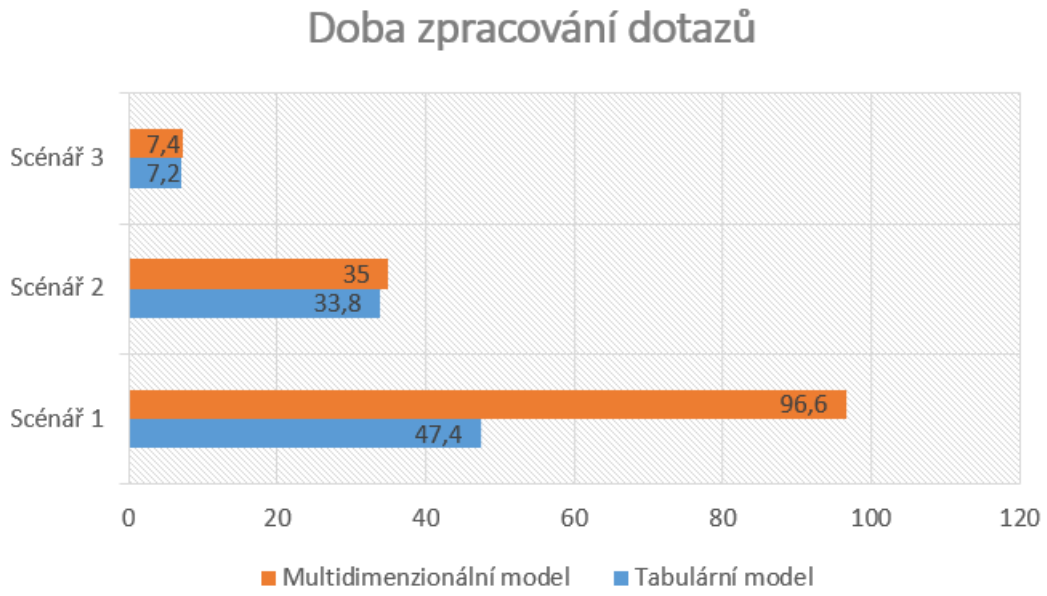
8.6 Shrnutí

Celkem bylo provedeno deset iterací pro každý scénář. I přes počáteční úmysl provést mnohonásobně více iterací se toto ukázalo jako zbytečné díky nepříliš velké variabilitě mezi jednotlivými výsledky. Ukázalo se, že pět iterací pro jednotlivé modely může poskytnout dostatečně přesné výsledky. Zde je vhodné poznamenat, že po každé iteraci byla vyprázdněna cache (Příloha E). V opačném případě by totiž opakované provádění dotazu z naplněné cache paměti přineslo velice rychlé, avšak nepříliš vypovídající výsledky.

V prvním scénáři se ukázalo použití tabulárního modelu jako nejvýhodnější, což bylo umožněno spojením agregační funkce SUM s další agregační funkcí COUNT DISTINCT. Dalším faktorem byla nízká variabilita dat ve sloupci produktů. Dimenze produktů totiž na rozdíl od faktové tabulky prodejů zůstala v nezměněné (nerozšířené) podobě. Z těchto faktů byl schopen tabulární model vytěžit přibližně dvojnásobný výkon oproti multidimenzionálnímu a lze tedy usoudit, že při podobných požadavcích na model by mohla být volba tabulárního modelu tou správnou.

V dalších dvou scénářích dosáhly oproti očekávání oba modely velice podobných výsledků. To bylo zapříčiněno použitím sloupců s vyšší variabilitou dat, tedy použitím převážně sloupců obsahujících finanční údaje (jako jsou například ceny, náklady atd.) z rozšířené faktové tabulky prodejů. V takovém případě je kompresní algoritmus xVelocity enginu méně účinný. Je také nucen vytvořit větší slovník, s čímž roste i počet vstupně-výstupních (I/O) operací, což má za následek zvýšení času nezbytného k vykonání dotazu.

Grafický souhrn výkonnosti jednotlivých modelů lze vidět v tomto grafu:



Obrázek 13: Doba zpracování dotazů v tabulárním a multidimenzionálním modelu (Zdroj: vlastní práce)

9 Ekonomické zhodnocení

S výběrem použitých technologií u projektu se vždy pojí i nutnost porovnat tyto technologie z ekonomického hlediska. Je nutné řešit otázku, jak splnit požadavky zákazníka s minimálními náklady na projekt. Ani výběr analytického modelu není výjimkou. S volbou tabulárního, respektive multidimenzionálního modelu souvisí i potenciální ztráta, či úspora finančních prostředků.

Pořizovací náklady na MS SQL Server 2012 jsou rozdílné v závislosti na zakoupené verzi produktu. Jak bylo psáno výše, tabulární model je podporován pouze ve verzi Enterprise a Business Intelligence. Multidimenzionální model je naproti tomu dostupný již v základní verzi Standard. Náklady na pořízení jednotlivých verzí jsou následující (Fontecchio, 2014):

Tabulka 14: Náklady na pořízení MS SQL Server 2012 [USD]

	Standard	**Business Intelligence**	**Enterprise**
Server licence	734	7026	-
CAL licence (min. 5ks)	207	207	-
CPU licence	6874	-	6874

Do pořizovacích nákladů je potřeba vzít v úvahu kromě softwarových také hardwarové prostředky. Náklady na RAM paměť, která je u tabulárního modelu kritická, jsou vyšší, než náklady na diskový prostor, do kterého jsou data ukládána u multidimenzionlního modelu. Pro srovnání lze uvést cenu 128 GB RAM paměti – 2326 dolarů (Kingston, 2014) a korespondujícího diskového prostoru, řekněme 960 GB SSD – 472 dolarů.

Multidimenzionální model bude, alespoň tedy zezačátku vývojového cyklu BI řešení, levnější variantou. To však neplatí pro samotnou tvorbu analytického modelu. Jak uvádí přední odborníci (Russo, Webb, Ferrari, 2012), vytvoření tabulárního modelu je až několikanásobně časově úspornější, než vytvoření multidimenzionálního modelu. To se potvrdilo i při vytvoření analytických modelů pro účely této práce. Vytvoření tabulárního modelu pro účely této práce trvalo autorovi bez předchozích zkušeností 25 hodin. Vytvoření ekvivalentního multidimenzionálního modelu naproti tomu trvalo zhruba 55 hodin. Tabulární model je pro uživatele snazší a intuitivnější, nabízí méně možností, proto bude i v této fázi projektu ekonomicky výhodnější.

Totéž bude platit i pro vytváření reportů. U klasického OLAP řešení k tomu můžeme využít Reporting Services, případně PowerPivot a podobně. Tabulární model však exkluzivně nabízí nový vizualizační nástroj PowerView, se kterým je vytvoření dashboardu otázkou několika hodin. Dle autorů (Russo, Webb, Ferrari, 2012) je možno tímto způsobem ušetřit až třetinu času nutného pro tvorbu reportů.

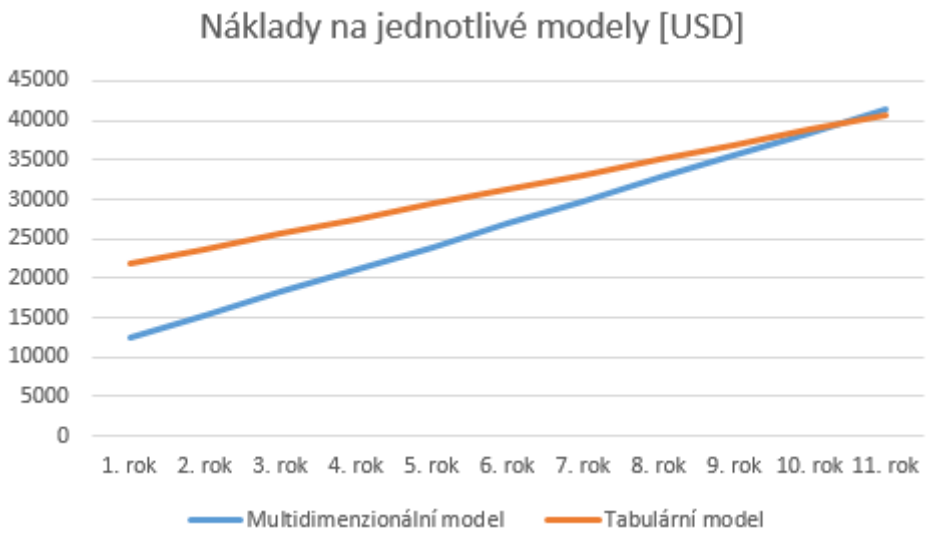
Správa a údržba modelu opět vychází lépe pro Tabular a to zejména pro jeho nižší komplexitu a uživatelskou přívětivost.

Pokud vezmeme v úvahu modelovou situaci, stavějící na empirických poznatcích při tvorbě praktického výstupu této diplomové práce, náklady na první rok BI projektu by se daly znázornit do následující tabulky:

Tabulka 15: Náklady v prvním roce BI projektu [USD]

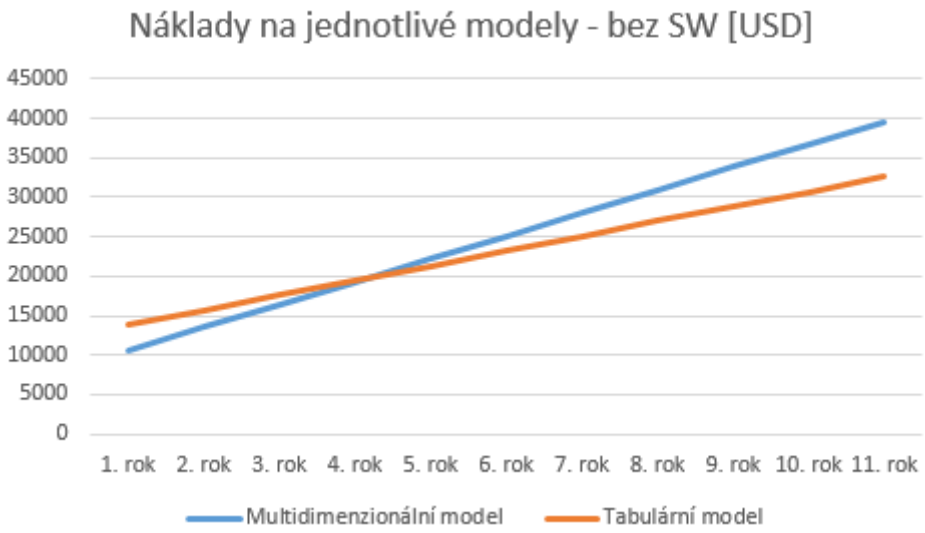
	Multidim. model	**Tabulární model**
Pořízení licence na software	1769	8061
Pořízení hardware	3472	5326
Vývoj analytického modelu	2500	5500
Tvorba reportů	1750	1100
Správa a údržba BI řešení	2900	1880
Celkem	12391	21867
Celkem bez SW	10622	13806

V případě, že firma nemá zakoupenou licenci na MS SQL Server 2012 Enterprise, případně BI edition, promítne se tato částka poměrně výrazně do celkových nákladů. V takovém případě by se tabulární projekt vyplatil až po jedenácti letech, jak lze vidět na následujícím grafu:



Obrázek 14: Náklady na BI projekt v čase – včetně zakoupení licence na SW (Zdroj: vlastní práce)

Pokud však firma již nakoupené licence má, tabulární řešení se jí vyplatí již mezi čtvrtým a pátým rokem, viz obr 15.



Obrázek 15: Náklady na BI projekt v čase – bez zakoupení licence na SW (Zdroj: vlastní práce)

Dále by ve prospěch tabulárního modelu mohl hrát například fakt, že čas nutný k vyškolení IT pracovníka bude nižší, díky nižší složitosti a vyšší intuivitě tohoto modelu. Rozsah znalostí takového pracovníka může být menší, s čímž se také pojí nižší platové ohodnocení a další úspora firmy. Volba modelu analytické databáze by tak měla být provedena velmi pečlivě.

10 Zhodnocení a závěr

Po vytvoření tabulárního a multidimenzionálního Business Intelligence modelu a srovnáním obou variant z pohledu vývoje, nasazení a výpočetní rychlosti, lze vyhodnotit výsledky diplomové práce.

10.1 Přínosy diplomové práce

V teoretické části práce byl představen pojem Business Intelligence, vysvětlen sémantický model BI a definovány trendy s ním související, provedena studie v současnosti využívaných BI nástrojů firmy Microsoft. Dále byl podrobně rozebrán Tabular, jakožto nový „in-memory“ databázový model a přidružený engine xVelocity. V neposlední řadě bylo uvedeno také srovnání těchto modelů společně s řadou doporučení, který model za jaké situace vybrat s ohledem na jejich funkcionalitu a vlastnosti.

V praktické části práce jsou pak tyto teoretické koncepty uplatněny a demonstrovány na ukázkovém BI řešení. Toto BI řešení bylo kvůli nárokům na výpočetní techniku nasazeno na virtuální server, aby bylo možno prozkoumat především kvantitativní vlastnost analytické databáze jednotlivých modelů – výpočetní rychlost. Tato vlastnost je extrémně důležitá především u tzv. real-time[13] systémů, kdy odezva na dotazy prováděné nad analytickou databází často znamená získání, či ztrátu zákazníka. Testování proběhlo pomocí testovacích scénářů, které simulují běžné požadavky firmy při analýze dat.

Diplomová práce by mohla být zvlášť přínosná pro někoho, kdo se rozhoduje mezi jednotlivými analytickými modely, a i přes velký objem zpracovávaných dat není komplexita modelu vysoká, či projekt nevyžaduje speciální funkcionalitu v podobě prvků, které poskytuje pouze multidimenzionální model. Také může být zajímavá pro kohokoli, kdo považuje rychlost odezvy za velmi důležitou, například IT oddělení implementující výše zmíněné real-time systémy. Práce poskytuje ucelený obraz o rozdílném přístupu k tvorbě sémantického modelu a lze se podle ní rozhodnout, zda zvolit tabulární, či multidimenzionální model v konkrétních situacích.

10.2 Diskuze

Jelikož je tabulární model poměrně novou záležitostí, dá se říci, že tato práce je unikátním počinem ve srovnávání jednotlivých modelů SQL Serveru. Přesto, že se v posledních letech zabývá mnoho knih a odborných prací tvorbou tabulárního modelu, málo z nich si vytyčilo za cíl skutečně ukázat jeho sílu a potenciál. Již publikované práce se zpravidla dělí na dva typy. První z nich pojednává o tvorbě modelu, jakožto součásti samoobslužného BI řešení v podobě programu Excel s nástavbou PowerPivot, případně jeho propojení se službou SharePoint u týmového BI. Druhým typem

[13]U Real-time systémů je důležité, aby při práci s daty poskytovaly tyto systémy velmi rychlou odezvu. Dle typu systému se tato odezva může pohybovat v rámci milisekund, ale případně i mikrosekund, či sekund.

je pak tvorba analytické databáze v SQL Serveru (Analysis Services + Data Tools), přičemž úkolem je především ukázat, že tabulární model je pouze jakýmsi zjednodušením při vývoji BI řešení. Použití tabulárního modelu se však může vyplatit i u rozsáhlých projektů, což se snaží tato práce dokázat. I když se jedná o poměrně neprobádaný prostor, o totéž se pokusil jeden z předních odborníků na tabulární model – A. Ferrari – při jeho komerčním nasazení pro firmu zabývající se cloudovými kontaktními centry, které se později ukázalo jako úspěšné (Ferrari, 2014). Snaha autora této diplomové práce bylo vyrovnat se kvalitě BI řešení poskytnutého těmito odborníky. Přestože nebyly odzkoušeny všechny varianty zmíněné v uvedeném white-paperu kvůli omezeným prostředkům virtuálního serveru, byly oproti řešení A. Ferrariho navrženy navíc testovací scénáře, které zvyšují důvěryhodnost výsledků práce.

10.3 Možnosti rozšíření

Přesto, že práce poskytuje ucelený vhled do problematiky tvorby analytického modelu databáze v SQL Serveru, bylo by možné ji rozšířit o několik prvků. Tím je například odlišná konfigurace virtuálního serveru, která by se mohla významně projevit na kvantitativních výsledcích práce. Jedním z těchto parametrů je konfigurace tzv. NUMA[14], což by potenciálně při správném nastavení znamenalo zvýšení výkonu databáze a snížení odezvy, kterou generují dotazy, které se nad touto databází provádějí.

Jedním z klíčových faktorů při výkonnosti BI řešení je návrh samotného modelu a empirické poznatky z dalšího testování různé míry normalizace tabulek a jejich struktury, partitioningu, optimalizace dotazů a podobně, které by mohly mít pozitivní vliv na odezvu a náročnost na systémové prostředky.

Dalším z rozšíření by mohlo být vytvoření dalších testovacích scénářů, které by mohly ukázat tabulární model jakožto výhodnější variantu.

10.4 Závěr

Cílem této práce bylo demonstrovat možnosti využití tabulárního režimu při tvorbě datových skladů na základě teoretických konceptů stojících za „in-memory“ technologiemi. Tohoto cíle bylo naplněno ve dvou hlavních částech práce.

V první části práce byly podrobně popsány teoretické koncepty tabulárního režimu a vysvětleny principy, na kterých tento model pracuje. Z textu lze také vyčíst, jaké přidružené technologie a nástroje tabulární model využívá. Teoretická část práce také obsahuje informace o klasickém multidimenzionálním modelu, přičemž

[14]Non-Uniform Memory Access (též Non-Uniform Memory Architecture, NUMA) je v informatice specifická počítačová platforma, která dosahuje škálovatelnosti tak, že seskupuje procesory a operační paměť do jednotek (neboli uzlů), které pak mohou samostatně fungovat téměř jako jeden počítač. (Finn, 2013)

jej srovnává s tabulárním z několika hledisek tak, aby bylo zřejmé, jak který model pracuje, co nabízí, a v čem jsou výhodnější jednotlivé modely. Nakonec práce čtenářovi předkládá v této části srovnání prvků obou modelů a nabízí sérii doporučení, kterými se lze řídit při výběru databázového modelu. Text je doplněn vlastními obrázky autora, případně ilustrativními obrázky z odborné literatury.

Teoretický aparát pak přechází v praktickou demonstraci tabulárního režimu. Postupně je popsán využitý datový vzorek a jeho struktura. Následně je čtenář proveden jeho transformací v datový model a nejdůležitějšími fázemi tohoto procesu. Vždy je uvedeno srovnání s multidimenzionálním modelem, aby bylo zřejmé, v čem spočívají odlišnosti obou modelů.

Aby bylo možné dokázat, že potenciál tabulárního režimu nespočívá pouze v jednoduchosti a rychlosti jeho použití, poslední kapitoly práce se věnují jeho využití u rozsáhlého modelu s více než miliardou záznamů, který slouží jako alternativa klasického OLAP modelu, který je běžně využívaný pro projekty tohoto rozsahu. Z výsledků výzkumu této práce vyplývá, že můžou nastat situace, kdy je pro určitý typ BI projektů se specifickými požadavky a scénáři vhodnější a výhodnější nasadit tabulární model. Proto je možno říci, že bylo dosaženo vytyčeného cíle.

V současnosti je však oblast Business Intelligence velice dynamickým odvětvím informačních technologií a je nezbytné tuto evoluci bedlivě sledovat, neboť se mění každým dnem. Toho je důkazem mimo jiné i fakt, že v době tvorby této práce proběhlo vydání SQL Serveru 2014 od firmy Microsoft, jenž adresuje některé z nedostatků jeho předchozí verze, které jsou zmíněny v této práci.

11 Reference

[1] Cancel, D. *Data-Driven Startups.* Online, 2010
Dostupné na: http://davidcancel.com/data-driven-startups/ .

[2] Collie, R. *DAX Formulas for PowerPivot the Excel Pro's Guide to Mastering DAX: Analysis Services with MDX and DAX.* Holy Macro! Books, 2012
ISBN 9781615473328.

[3] Dewald, B. *SQL Server Analysis Services 2012 Cube Development Cookbook.* Packt Publishing Ltd, 2013
ISBN 9781849689816.

[4] Feldman, D. *Developing Business Intelligence Apps for SharePoint.* O'Reilly Media, Inc., 2013
ISBN 9781449324681.

[5] Ferrari, A. *Data-Driven Startups.* Online, 2014
Dostupné na: http://msdn.microsoft.com/en-us/library/dn751533.aspx .

[6] Finn, A. *Windows Server 2012 Hyper-V Installation and Configuration Guide.* John Wiley and Sons, 2013
ISBN 9781118651438.

[7] Harinath, S. *Professional Microsoft SQL Server 2012: Analysis Services with MDX and DAX.* John Wiley and Sons, 2012
ISBN 1118262093.

[8] Inmon, W. *Building the data warehouse.* John Wiley and Sons, 2005
ISBN 0764599445.

[9] Jorgensen, A. *Microsoft SQL Server 2012 Bible.* John Wiley and Sons, 2012
ISBN 9781118282175.

[10] Kimball, R. *The data warehouse toolkit: the complete guide to dimensional modeling.* John Wiley and Sons, 2002
ISBN 0471200247.

[11] Kingston *ValueRAM Server Memory* Online, 2014
Dostupné na: http://www.kingston.com/us/memory/valueram/server .

[12] Lachev, T. *Applied Microsoft SQL Server 2011 analysis services: tabular modeling.* Prologika Press, 2005
ISBN 9780976635352.

[13] Lobel, L. *Programming Microsoft SQL Server 2012.* O'Reilly Media, Inc., 2012
ISBN 9780735675285.

[14] Novotný, O., Pour, J., Slánský, D. *Applied Microsoft SQL Server 2012 analysis services: tabular modeling.* Prologika Press, 2012
ISBN 9780976635352.

[15] Russo, M., Ferrari, A., Webb, C. *Microsoft SQL Server 2012 analysis services: the BISM Tabular Model.* Prologika Press, 2012
ISBN 0735658188.

[16] Sabherwal, R. *Business Intelligence.* John Wiley and Sons, 2010
ISBN 9780470461709.

[17] Scheps, S. *Business Intelligence For Dummies.* John Wiley and Sons, 2013
ISBN 9780470262207.

[18] Schiller, M. *Co se skrývá pod zkratkou ETL?* Online, 2003
Dostupné na: http://www.systemonline.cz/clanky/co-se-skryva-pod-zkratkou-etl.htm .

[19] Singh, M. *Pro SharePoint 2013 Business Intelligence Solutions.* Apress, 2013
ISBN 9781430258940.

[20] Te Braak, P. *Microsoft Tabular Modeling Cookbook.* Packt Publishing Ltd, 2013
ISBN 9781782170884.

[21] Thomsen, E. *OLAP Solutions – Building Multidimensional Information Systems.* John Wiley and Sons. 2. vyd. 2002.
ISBN 0471400300.

[22] Vercellis, C. *Business intelligence data mining and optimization for decision making.* John Wiley and Sons, 2013
ISBN 9781119965473.

[23] Warren, N. *Business Intelligence in Microsoft SharePoint 2013.* Pearson Education, 2013
ISBN 9780735675872.

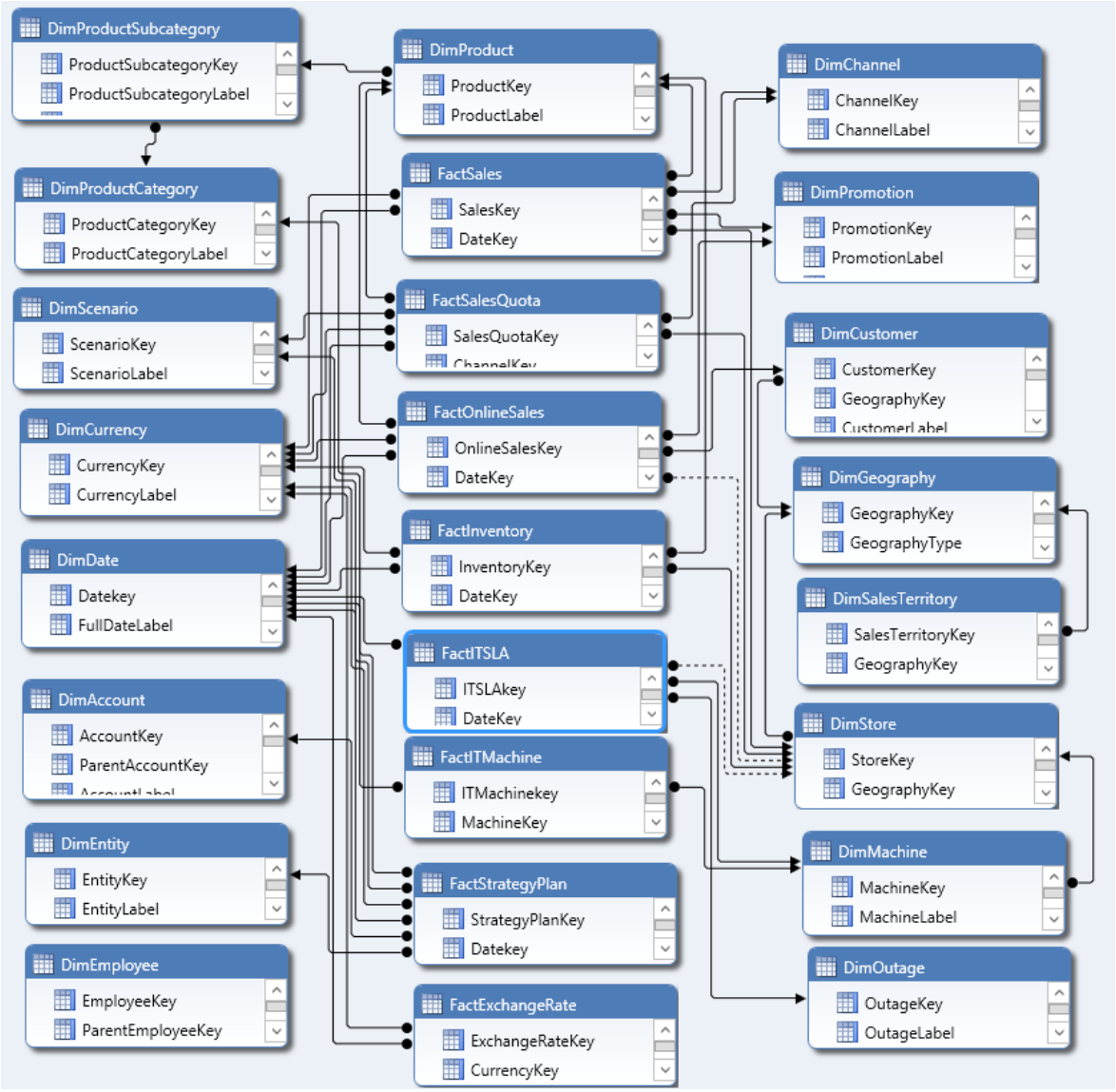
Přílohy

A T-SQL Skript 1

Zdrojový kód 11: T-SQL Skript pro rozšíření faktové tabulky Sales

```
1  DECLARE @maxstore INT
2  DECLARE @maxproduct INT
3  DECLARE @counter INT
4
5  SELECT @maxstore = MAX([StoreKey]) FROM [ContosoRetailDW].[dbo↩
       ].[DimStore]
6  SELECT @maxproduct = MAX([ProductKey]) FROM [ContosoRetailDW].[↩
       dbo].[DimProduct];
7
8  SET @counter = 0
9
10 WHILE @COUNTER < 353
11 BEGIN
12 SET @counter = @counter + 1
13
14 INSERT INTO [ContosoRetailDW].[dbo].[FactSalesBig]
15     ([DateKey]
16       ,[StoreKey]
17       ,[ProductKey]
18       ,[UnitCost]
19       ,[UnitPrice]
20       ,[SalesQuantity]
21       ,[ReturnQuantity])
22 SELECT dateadd(dd,(Abs(Checksum(NewId())) % 700 + 1),[DateKey])↩
       DateKey
23       ,(Abs(Checksum(NewId())) % 310  + 1) StoreKey
24       ,(Abs(Checksum(NewId())) % @maxproduct + 1) ProductKey
25       ,dp.UnitCost UnitCost
26       ,dp.UnitPrice UnitPrice
27       ,(Abs(Checksum(NewId())) % 25 + 1) SalesQuantity
28       ,(Abs(Checksum(NewId())) % 2) ReturnQuantity
29   FROM [ContosoRetailDW].[dbo].[FactSales] fs
30   JOIN ContosoRetailDW.dbo.DimProduct dp ON fs.ProductKey =  dp↩
       .ProductKey
31
32 END
```

B Schéma databáze Contoso



Obrázek 16: Diagram databáze Contoso (Zdroj: vlastní práce)

C Dotaz v jazyce MDX pro jednotlivé scénáře

Zdrojový kód 12: Prodeje produktů v předvánočním období (MDX)

```
SELECT NON EMPTY
   { [Measures].[Product Key Distinct Count],
     [Measures].[Customer Key Distinct Count],
     [Measures].[Store Key Distinct Count],
     [Measures].[Sales Amount]
   } ON COLUMNS,
NON EMPTY { ([Date].[Date].[Date].ALLMEMBERS ) } ON ROWS
FROM (
    SELECT ( [Date].[Date].&[20071201] : [Date].[Date].&[20071223]
    )
ON COLUMNS FROM [Operation])
```

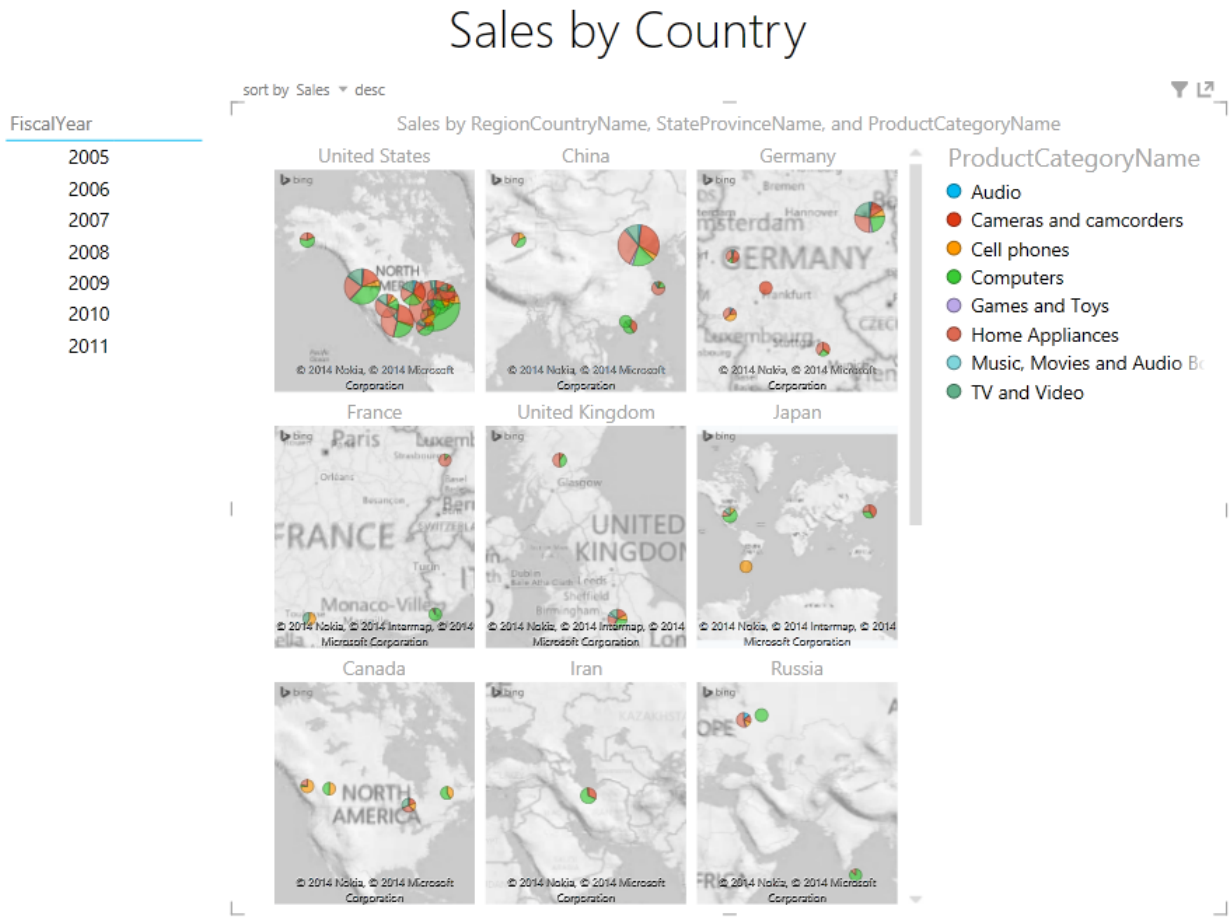
Zdrojový kód 13: Počet zákazníků v jednotlivých městech (MDX)

```
SELECT NON EMPTY { [Measures].[Customer Key Distinct Count] } ON COLUMNS,
NON EMPTY FILTER (
            [Geography].[City Name].[City Name].ALLMEMBERS,
            [Measures].[Customer Key Distinct Count]>100
           ) ON ROWS
FROM [Operation]
```

Zdrojový kód 14: Počet zákazníků za poslední rok u jednotlivých poboček (MDX)

```
SELECT NON EMPTY { [Measures].[Sales Amount] } ON COLUMNS,
NON EMPTY {
             ([Employee].[First Name].[First Name].ALLMEMBERS *
              [Employee].[Last Name].[Last Name].ALLMEMBERS )
          }
ON ROWS
 FROM [Operation]
```

D PowerView Report 1



Obrázek 17: Geografické zobrazení prodejů dle státu – PowerView(Zdroj: vlastní práce)

E XML skript pro vyprázdnění cache paměti

Zdrojový kód 15: Skript pro vyprázdnění cache paměti (XML)

```
1 <ClearCache xmlns="http://schemas.microsoft.com/↩
    analysisservices/2003/engine">
2   <Object>
3     <DatabaseID>TABULAR</DatabaseID>
4   </Object>
5 </ClearCache>
```