Česká zemědělská univerzita v Praze

Provozně ekonomická fakulta

Katedra systémového inženýrství

Business Intelligence

Diplomová práce

Autor: Bc. Ondřej Moravec

ČESKÁ ZEMĚDĚLSKÁ UNIVERZITA V PRAZE

Provozně ekonomická fakulta

ZADÁNÍ DIPLOMOVÉ PRÁCE

Ondřej Moravec

Informatika

Název práce

Business Intelligence

Název anglicky

Business Intelligence

Cíle práce

Cílem práce je vytvořit souhrn pravidel tvorby firemního reportingu, demonstrovat jednotlivé fáze tvorby reportu a jejich rizika a na závěr prakticky provést celý proces tvorby reportu v prostředí Oracle BI Inteligence, od analýzy požadavku, přes sběr dat v produkční databázi, transformaci dat do struktur datového skladu a následnou vizualizaci a prezentaci reportu.

Metodika

Vytvoření souhrnu pravidel pro tvorbu firemního reportingu bude založeno na syntéze znalostí prostředí Oracle BI získaných školením od specialistů, studiem odborné literatury a vlastních zkušenostech získaných při tvorbě firemního reportingu. V praktické části hodlám demonstrovat kompletní proces tvorby firemního reportu v prostředí Oracle BI, celý proces tvorby reportu zachytit vhodnými diagramy Business Process Model and Notation. Závěr diplomové práce shrne pravidla tvorby reportu v konfrontaci s tvorbou konkrétního reportu.

Doporučené zdroje informací

E-H.Khan, C. Screen, Oracle Business Intelligence, Packt Publishing Limited 2012, ISBN: 1849685665

Jana Fibírová, Reporting: moderní metoda hodnocení výkonnosti uvnitř firmy, Grada 2003, ISBN:9788024704821

Jan Pour, Miloš Maryška, Ota Novotný, Business intelligence v podnikové praxi, Professional Publishing 2012, ISBN: 978-80-7431-065-2

Ota Novotný, Jan Pour, David Slánský, Business Intelligence: Jak využít bohatství ve vašich datech, Grada Publishing a.s. 2004, ISBN: 80-247-1094-3

Robert Stackowiak, Joseph Rayman, Rick Greenwald, Oracle Data Warehousing and Business Intelligence Solutions, Wiley 2007, ISBN: 978-0-471-91921-6

Singh Lather, Business Intelligence and Data Warehousing, Narosa Publishing House 2012, ISBN: 8184872127

Yuli Vasiliev, Oracle Business Intelligence: The Condensed Guide to Analysis and Reporting, Packt Publishing Limited 2010 ISBN: 184968118X

Garantující pracoviště

Katedra informačních technologií

Elektronicky schváleno dne 31. 10. 2014

Ing. Jiří Vaněk, Ph.D.

Vedoucí katedry

Elektronicky schváleno dne 11. 11. 2014

Ing. Martin Pelikán, Ph.D.

Děkan

V Praze dne 10. 02. 2016

Čestné prohlášení

Prohlašuji, že svou diplomovou práci "Business Inteligence" jsem vypracoval samostatně pod vedením vedoucího diplomové práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou citovány v práci a uvedeny v seznamu literatury na konci práce. Jako autor uvedené diplomové práce dále prohlašuji, že jsem v souvislosti s jejím vytvořením neporušil autorská práva třetích osob

V Praze dne 15. 2. 2016 _____

Poděkování

Rád bych touto cestou poděkoval Ing. Janu Tyrychtrovi, Ph.D. za odborné vedení a pomoc při zpracování diplomové práce.

Souhrn

Práce je věnována problematice implementace Business Intelligence v podnikovém prostředí. Nejprve je vysvětlen samotný pojem Business Intelligence, dále jsou uvedeny nejnovější trendy v oblasti Business Intelligence, možnosti současných moderních reportingových nástrojů BI pro zpracování dat a jejich prezentaci. Hlavní část teoretické části je věnována souhrnu pravidel tvorby firemního reportingu, při tomto jsou popsány jednotlivé fáze vývoje reportu včetně rizik, které jsou s těmito fázemi spojeny. V praktice části je proveden kompletní vývoj reportu v prostředí Oracle BI.

Klíčová slova

Business Intelligence, databáze Oracle, firemní reporting, datový sklad, Oracle BI server, best practices.

Summary

Work is devoted to the implementation of Business Intelligence in the corporate environment. First, an explanation very concept of Business Intelligence, hereinafter referred latest trends in Business Intelligence, the possibilities of the modern BI reporting tools for data processing and presentation. The main part of the theoretical part is devoted to a set of rules making corporate reporting, in this describes the different stages of development report including the risks that are associated with these phases. The practitioners often is made complete development report in Oracle BI.

Keywords

Business Intelligence, Oracle database, Corporate reporting, Data Warehouse, Oracle BI Server, best practices.

Obsah

1. Úvod

V oblasti Business Intelligence, neboli BI, dochází ke spojení znalostí světa techniky a světa podnikání. Díky nejrůznějším aplikacím lze využít podniková nebo veřejně přístupná data pro vyhodnocení dosavadního vývoje podnikatelské činnosti a stejně tak vytvořit předpovědi do budoucna o možných příležitostech a ohroženích.

Pro řízení většiny společností je díky tomuto využití Business Intelligence nepostradatelné. Proto jsou vyvíjeny stále dokonalejší nástroje pro získávání, ukládání, transformaci a prezentaci dat, které napomáhají k tvorbě firemního reportingu. Jakkoliv je vývoj těchto nástrojů dynamický, ostatní z faktorů úspěšné tvorby firemního reportingu zůstávají až konstantně neměnné. Těmito faktory jsou zejména snaha o čistotu dat v primárních systémech, kvalitní návrh a dodržení základních pravidel vývoje software doplnění o specifika vývoje reportu.

V odborných knihách věnujících se vývoji Business Inteligence je uvedeno mnoho případů, kdy velice nákladné projekty skončily nezdarem, a to i přesto, že se na vývoji podílely renomované vývojářské firmy. Vyvinuté reporty neodpovídaly představám zadavatelů nebo jimi nebyly přijaty a užívány, takže původní záměr dodat svým zaměstnancům nový nástroj pro podporu jejich pracovní činnosti skončil vysokými náklady na řešení, které své společnosti nepřineslo žádný nebo velmi omezený užitek. Přestože se však jednalo o řešení na různých platformách, příčiny těchto selhání se opakovaly.

Dodržování zásad vývoje firemního reportingu je tedy zásadním faktorem, který rozhoduje o úspěšnosti projektu nebo o zmaru vynaložené investice.

2. Cíl práce a metodika

2.1. Cíl práce

Cílem diplomové práce je sestavit základní pravidla tvorby reportu v podnikovém prostředí. Při tomto se nejedná o pokus vytvoření komplexní metodiky, spíše o vytvoření základního vývojářského minima k vyhnutí se těm nejběžnějším chybám, které nejvíce ohrožují úspěšné vytvoření reportu, který bude jeho zadavateli akceptován a využíván.

Dílčím cílem je stanovení rizik v jednotlivých fázích vývoje reportu a zvážení postupů, které by tato rizika měla minimalizovat.

V praktické části bude předveden vývoj jednoduchého reportu v prostředí Oracle BI, jednotlivé kroky postupu budou konfrontovány se zásadami uvedenými v části.

2.2. Metodika

Pro diplomovou práci byla zvolena metodika analýzy odborných zdrojů, ale pak syntéza zjištěných informací se znalostí prostředí Oracle BI získaných školením od specialistů a osobními pracovními zkušenostmi při tvorbě firemního reportingu.

Pro popis jednotlivých fází vývoje reportu byl jako metodický Framework vývoje software zvolen vodopádový model. U jednotlivých fází tvorby reportu budou uvedena jejich rizika a možná řešení, která by tato rizika měla eliminovat.

V praktické části bude proveden kompletní vývoj reportu v prostředí Oracle BI Inteligence, od analýzy požadavku, přes sběr dat v produkční databázi, výběr, fyzické uložení dat a transformaci dat do struktur datového skladu a následnou vizualizaci a prezentaci reportu. Vzhledem k rozmanitosti procesu vývoje bude obrazová dokumentace zařazena přímo do diplomové práce, aby obrázky co nejvíce dokreslovaly aktuálně popisovanou část postupu. Vzhledem k obsáhlosti popisu praktické části bude vytvořen pouze jediný report, neboť vytváření plnohodnotného panelu s více reporty by přesáhlo požadovaný rozsah.

Pro vývoj reportu budou užity tyto komponenty BI:

- Vývojová a testovací databáze Oracle 10.2.0.5. g, která je instalována na vyhrazeném serveru s operačním systémem Centos 6.5.

  V databázi budou za použití jazyka SQL vytvořeny potřebné tabulky ve schématu BI, aby ETL nástroj nebyl přímo napojen na zdrojovou tabulku a jeho činnost neohrozila činnost firemní databáze. Z produkční databáze budou extrahována data ze zdrojových tabulek do vrstvy STG0 datového skladu v poměru 1:1.

- Produkční databáze Oracle 10.2.0.5. g, která je instalována na vyhrazeném serveru s operačním systémem Centos 6.5.

  V produkční databázi bude nasazeno řešení po úspěšném ukončení testovací fáze na vývojové a testovací databázi.

- ETL nástroj Kettle Pentaho 6.0. – vývojové a testovací prostředí

  Nástroj bude použit pro výběr dat z produkční databáze a fyzické uložení do datového skladu, které bude uskutečněno tzv. jobem aplikace, založeném na skutečné databázové úloze. Uvedený nástroj byl vybrán z důvodu, že je open source a na rozdíl od databázového linku, se kterým by bylo možné provést stejnou činnost, nástroj Kettle disponuje transformacemi,

které zajišťují zaznamenání a oznámení chyby, opakovaný pokus o provedení jobu.

- ETL nástroj Kettle Pentaho 6.0. – produkční prostředí

V nástroji bude nasazen otestovaný proces ve vývojovém a testovacím prostředí

- Datový sklad, vývojové a testovací prostředí, databáze Oracle 10.2.0.5. g instalovaná na vyhrazeném serveru s operačním systémem Windows Server 2012 s diskovým polem 2 TB.

  Datový sklad se skládá ze 3 logických částí, vrstev STG0, STG1 a STG2. Vrstva STG0 slouží jako úložiště dat z produkční databáze v poměru 1:1. Všechny tabulky ve vrstvě STG0 obsahují navíc pracovní sloupec row_creation_date, který obsahuje datum, kdy se uvedený snímek naplnil. Protože při zpracování dat může dojít k chybám a následným pádům, tabulky ve vrstvě STG0 vždy obsahují 5 snímků – tedy data za posledních 5 dnů, aby v případě potřeby bylo možné zopakovat plnění cílových tabulek datového skladu. Po úspěšném naplnění nového snímku se vždy smaže nejstarší snímek. Vrstva STG1 obsahuje tabulky, kde jsou data z původních tabulek vyčištěná od datových nekonzistencí, prázdné hodnoty, které jsou nahrazovány hodnotou N/A, může obsahovat i tabulky, které vznikly spojením dat z různých tabulek ve vrstvě STG0. Vrstva STG2 obsahuje specializovaná datová tržiště, která vznikají z tabulek ve vrstvě STG1, často jsou vysoce agregovaná.

  Při transformaci a agregaci dat do vrstvy STG1 bude použita procedura v jazyce SQL, v produkční databázi neexistuje vazba mezi platbou a exekutorskou společností, tato vazba bude vytvořena na základě vyhodnocení data připsání platby na účet inkasní společnosti a období, kdy měla uvedený spis ve správě daná exekutorská společnost.

Při agregaci dat do vrstvy STG2 datového skladu bude opět použita procedura SQL, zde již bude agregace minimální, ale vzhledem k tomu, že data ve zdrojových tabulkách jsou často opravována pro chyby uživatelů při vkládání, bude vždy cílové datové tržiště smazáno a naplněno celé. Užití procedur při plnění vrstvy STG1 je náročné na prostředky databáze, proto během jejího plnění bude datové tržiště na vrstvě STG2 stále dostupné uživateli a jeho smazání a znovunaplnění proběhne do tří sekund.

- Datový sklad, produkční prostředí, databáze Oracle 10.2.0.5. g instalovaná na vyhrazeném serveru s operačním systémem Windows Server 2012 s diskovým polem 2 TB.

- Manažerská aplikace - Analytický a reportingový nástroj Oracle BI 10 g, testovací a vývojové prostředí, ze kterého v práci budou využity uvedené moduly:

Oracle BI server - obsahuje logický datový model, kde jako zdroje jsou konkrétní datové soubory. Metadata modelu jsou uložena v souboru rpd, který je nazýván Repository. Přístup k repository je možný přes Oracle BI Administration Tool.

V praktické části diplomové práce budou prostřednictvím uvedeného nástroje mapované fyzické tabulky datového skladu ve vrstvě STG, aby dále mohly být použity pro tvorbu reportu. Mapování tabulek musí proběhnout na všech třech vrstvách modelu.

Na fyzické vrstvě dojde k importu metadat tabulky z datového skladu do repository a ke spojení s dimenzionálními tabulkami. Importovaná faktová tabulka bude obsahovat agregované záznamy o částkách a platbách exekučních případů, dále pak klíče dimenzionálních tabulek.

Na business vrstvě dojde k vytvoření hierarchie, která umožní následné použití operace drill-down v reportu, takže data ve faktové tabulce bude možné interaktivně zobrazovat a agregovat podle úrovně detailu požadované uživatelem.

V této vrstvě bude též nastaven typ agregace sloupců, které budou obsahovat faktová data.

Na presentační vrstvě modelu budou dimenzionální tabulky a faktová tabulka přidány do vhodné logické složky, toto přiřazení rozhoduje o přehlednosti zdrojů využitelných k tvorbě reportů a také o možnosti přístupu jednotlivých uživatelů k těmto zdrojům.

Answers – jedná se nástroj pro tvorbu reportů, který obsahuje složky z presentační vrstvy Repository i s faktovými a dimenzionálními tabulkami. V Answers bude proveden výběr vhodných sloupců z faktové tabulky a dimenzionálních tabulek pro report, výstupní formát agregovaných sloupců bude zformátován pro lepší čitelnost zaokrouhlením na celá čísla a oddělením tisíců. Dále bude vytvořena výzva panelů, jedná se o strukturu sdružující pole, které pak v rámci konkrétního reportu, stránky panelů nebo celého panelu slouží jako filtry.

Dashboards – jedná se o nástroj pro konečnou vizualizaci reportů. Architekturu Oracle Dashboards lze rozdělit na panely a stránky. Panel je objekt, který pod sebou může mít jednu a více stránek, na které se umisťují konkrétní reporty. Tato architektura umožňuje nastavení práv skupin nebo práv uživatelů na celý panel nebo jen některé ze stránek, které jsou na panelu umístěny. Práva se dají nastavit i na konkrétní report na stránce, zpravidla se ale tvoří stránky tak, aby obsahovaly jen reporty, které jsou všechny přístupné skupině nebo uživateli s právem na celou stránku.

- Manažerská aplikace - Analytický a reportingový nástroj Oracle BI 10 g, produkční prostředí. V aplikaci bude nasazen otestovaný report vyvinutý na vývojovém a testovacím prostředí.

3. Teoretická část

3.1. Základní pojmy

3.1.1. Firemní reporting

Úkolem reportingu, jako jedné z velmi důležitých částí controllingu, je vytvořit relativně komplexní systém ukazatelů a informací, které by měly vyhodnocovat nejen vývoj podniku jako celku, ale v takových dílčích částech a pohledech, které jsou z hlediska řízení rozhodující. (Fibírová,2003)

Toto ryze ekonomické pojetí slova pochází z doby, kdy byl reporting chápán jako nedílná součást controllingu, týkal se převážně finančních ukazatelů firmy a výsledky byly přístupné pouze úzkému okruhu osob v podniku. Oproti tomu Tyrychtr(2014) pod pojmem chápe reporting jako analytickou vrstvu, která je „zaměřená na standardní nebo ad hoc dotazovací proces, do databázových komponent řešení BI“.

Průnikem obou definic se dá charakterizovat širší pojetí významu slova reporting tak, jak je užíván v této diplomové práci. Podle Urbana (2013) se současný trend rozhodování ve společnostech se posunul směrem od shora dolů, o těch zásadních vizích, dlouhodobých a strategických cílech sice stále rozhodují vrcholní manažeři, současně s tím se ale odpovědnost za podobu a efektivitu podnikových procesů přesunula na střední a nižší management, až na řadové zaměstnance, což souvisí s vírou ve větší efektivitu v důsledku umožnění osobní iniciativy, kreativity a odpovědnosti.

Předmětem reportů již nejsou pouze finanční ukazatele, ale například i různé měřitelné kvantitativní a kvalitativní ukazatele podnikových procesů či preference a chování zákazníků. Za využití informačních technologií tvořeny a zobrazovány jako historické reporty dosavadního vývoje, prediktivní reporty vytvořené za použití sofistikovaných statistických funkcí a různá operativní hlášení a varování.

Vedení společnosti by tedy mělo systematicky podporovat tvorbu firemního reportingu, aby byl co nejvíce komplexní a pokrýval potřeby zaměstnanců.

3.1.2. Business Intelligence

Business Intelligence je zastřešující termín, který se vztahuje ke znalostem procesům, technologiím, aplikacím a postupům, které usnadňují podnikové rozhodování. Technologie Business Intelligence pracuje s použitými (historickými daty) v požadovaném kontextu a pomáhá přijímat podniková rozhodnutí pro budoucnost. (Lamberge, 2012)

Současným trendem je směr k takzvanému agilnímu BI, což znamená co okamžité a vždy dostupné využití informací a znalostí, tedy výstupů reportů.

Agilní BI umožňují uživatelům začít pracovat s daty mnohem rychleji a přizpůsobit je měnícím se potřebám. (Ralph Hughes, 2013).

Současně s tím vzrostlo množství příjemců reportů, příjemcem v podstatě může být každý zaměstnanec firmy, i když množství a důvěrnost informací je samozřejmě odstupňováno dle zaměstnaneckých rolí v podniku.

Pro agilní Business Intelligenci je tedy charakteristická mobilita výstupů, nové zdroje a forma dat. Jednotlivé charakteristiky trendu lze rozdělit na pojmy:

3.1.3. Self-service BI

Pod tímto pojmem se skrývá, jednoduchost, rychlost, intuitivní využití, aktivní zapojení cílového uživatele do tvorby vlastních reportů, scheduling, alerting, integrace

s finančními a ERP nástroji podniku. Snahou je minimalizovat investici času uživatele na nalezení reportu v místě zobrazení, osvojení si ovládání všech aktivních prvků reportu a hlavně pochopení toho, co uvedený report vlastně sděluje.

Samoobslužné BI aplikace a řešení (self-service BI) tak představují jeden z nejvýraznějších vývojových směrů v oblasti Business Intelligence. Jejich smyslem je na základě nových technologií poskytnout uživatelům prostředí pro realizaci svých analytických úloh bez nutnosti využívání komplexních a obvykle velmi složitých systémů BI. Samoobslužné BI umožňují např. realizovat multidimenzionální uložení a zpracování dat, nabízejí efektivní a jednoduché přístupy k datům, poskytují prostředky pro výpočty a další operace v dimenzionálním prostředí apod. Kromě toho jsou tyto aplikace i vhodným prostředkem pro pochopení podstaty a způsobu využití větších Business Intelligence systémů.(Pour, 2014)

Zapojení cílového uživatele do tvorby reportů je možné díky tomu, že tvůrci nástrojů Business Intelligence v současné době vytváří takzvaná enterprise řešení, v praxi to znamená, že zkušení tvůrci datových skladů systematicky vytvoří jednotnou, konzistentní datovou vrstvu, jejíž jednotlivé entity(většinou tabulky) jsou na určité vrstvě nástroje dostupné tvůrcům koncových reportů, což už v takovém případě nemusí být programátoři, ale mohou to být business analytici s výbornou znalostí podnikových procesů bez schopnosti napsat jediný řádek programového kódu. V praxi to pak znamená omezení potřeby alokace času vývojářů, kteří jsou pro podnik nákladní, pracovně přetížení a na trhu práce stále nedostatkoví. Reporty se tak do obsahu a vizualizace mohou co nejvíce přiblížit představě těch, kteří jejich vytvoření požadují.

Obrácenou mincí tohoto přístupu je zvýšené úsilí při dodržení pravidla jedné pravdy, které definoval Will Inmon - „otec datových skladů“, a dále pak zajištění bezpečnosti dat a informací.

Mobilní BI

V současné době je naprostou samozřejmostí schopnost data výstupu zobrazit na mobilních zařízeních a to nejen v prohlížečích, ale třeba i formou speciálních mobilních aplikací pro daný nástroj, které optimalizují zobrazení vzhledem zařízení příjemce nebo jeho preferencím.

Analýzy ukazují, že mobilní BI je k tradičnímu BI spíše komplementární, než jeho náhradou. Komplexní analýzy jsou prováděny v tradiční BI, zatímco od mobilního BI se očekávají jednoduché analýzy, možnost rozhodování kdykoliv a kdekoliv, redukce rozhodovacích času zejména v kritických situacích, zlepšení komunikace mezi rozhodovacími orgány, spolupráce s třetími stranami a lepší služby zákazníkům. (Tona, Carlsson, 2013)

Naopak data z mobilních zařízení mohou být i vstupem pro datové analýzy BI a nabídky v oblasti e-commerce. Proto se zvyšuje požadavek na zpracování dat v paměti (in-memory computing), což znamená co nejrychlejší zpracování a využití od jejich získání, protože cena dat se v takovém případě výrazně snižuje se zvětšující se dobou mezi získáním a zpracováním.

3.1.4. Big data

Jedná o pojem, který je v České republice v poslední době často skloňovaný na odborných fórech týkající se BI. Big data jsou o zpracování objemných dat, často nestrukturovaných, různého typu formátů a z různých zdrojů, jako jsou sociální sítě, e-maily či vyplněné komentáře formulářů od zákazníků a následné integraci například do CRM systémů. „Jádrem Big Dat jsou předpovědi na základě korelace“(Mayer-Schönberger, Cukier,2013).

Nadšení z této technologie bylo obrovské, například Craiga Mundie, Senior poradce generálního ředitele společnosti Microsoft, tvrdí, že „Data se stávají novou surovinou v podnikání."

Přestože uvedená technologie disponuje velkým potenciálem, dle oslovených manažerů a vývojářů projektů Big Dat z ČSOB, T-Mobile a O2 v České republice však výsledky dosud nenaplnily očekávání, například v O2 monetizace projektu Big Dat z roku 2012 skočila na třetinovém výsledku oproti plánu.

Potvrzením, že zde zmiňované případy nejsou vytržené z reality, je "Křivka humbuku" firmy Gartner, která zobrazuje různé fáze přijímání technologie. V roce 2014 byla Big Data na nejprudší hraně sestupu v zóně pocitu deziluze, v roce 2015 na křivce chybí úplně, což by mohlo znamenat, že technologie Big Data už si našla své pravé místo a je vnímána podle skutečného užitku.

Jenom vlastnictví a osvojení si principů technologie Big Dat tedy nestačí. Výstižné jsou v tomto směru výroky Merrylla(2012) že „Big Data nejsou o bitech, ale o talentu" a Mayer-Schönbergera(2013), že „získání informace není o tom, jak silné jsou stroje pro získávání dat, ale o myšlence."

3.1.5. Cloud

Stejně jako u jiných oblastí IT lze pro BI využít cloudová řešení. Podle Marešové (2015) je benefitem řešení Business Intelligence v cloudu snížení nákladů na IT specialisty, hardware, prostory, energie, klimatizaci a údržbu a softwarové licence. Jako rizika naopak vnímá možnost chyb, které nebude možné řešit prostřednictvím vlastních zaměstnanců okamžitě, dále pak riziko vyšších nákladů při implementaci systému a problémy s dostupností a bezpečností dat.

3.2. Obecné zásady tvorby firemního reportingu

V této kapitole budou uvedeny obecné zásady pro tvorbu firemního reportingu. Jejich nedodržení na počátku může způsobit zanedbatelné zrychlení vývoje, po určité době se však stává zdrojem nejasností a redundancí v reportovaných datech či reportech samotných.

3.2.1. Vypracování a zavedení firemního slovníku

Pokud různé skupiny v organizaci definují termín odlišně, nelze dosáhnout kvality dat. Je potřeba najít shodnu na podnikové terminologii, což znamená, že data je nutné rozdělit na jejich základní komponenty. Pro každou z těchto datových komponent i pro vlastní podnikovou položku je třeba vytvořit definici. (Lamberge, 2012)

Každý report by měl mít jednoznačnou vypovídající hodnotu. Zadání významu sloupce pro vývojáře by mělo být tak přesné natolik, že se v daném sloupci objeví požadované hodnoty a název sloupce by měl být pojmenován tak intuitivně, aby příjemce reportu nepřemýšlel nad tím, co konkrétní sloupec vlastně znamená nebo odkud se data pro uvedený sloupec vzala.

Mezi základní pilíře firemní kultury by tedy měla patřit standardizovaná firemní terminologie, aby při každém ústním nebo písemném jednání byl jednoznačný předmět jednání a jeho závěr. Tvorba firemního business slovníku musí být na úplném počátku tvorby firemního reportingu, protože pokud se odsune až za vývoj firemních reportů, stav na konci vývoje bude takový, že v různých reportech pro různě počítané hodnoty budou stejné názvy a naopak pro stejně počítané hodnoty budou různé názvy. Řešením poté bude reporty přepracovat a správně pojmenovat, což bude opět časově náročné nebo se smířit s uvedeným stavem – ani jedna z možností není správná a vyhovující.

Vytvoření, péči a používání datového slovníku mezi priority vedení společnosti, systematicky zaškolovat nové i stávající zaměstnance při nástupu do firmy, změnu pozice, vzniku nových procesů nebo fúzí společností. Jednotliví manažeři by v případě zjištění chybně používané terminologie sami měli iniciovat proškolení svých podřízených, ať se jedná o nováčky nebo dlouhodobé zaměstnance.

3.2.2. Kvalita dat primárních systémů

V současné době charakterizované enormním nárůstem objemu dat a silným tlakem na snižování nákladů je orientace na datovou kvalitu řešením, jak se s těmito protichůdnými tlaky vypořádat. Zyka (2012)

Aby data v produkční databázi byla použitelná jak pro potřeby firemních aplikací, tak následně jako zdroj datového skladu pro tvorbu reportů, musí být ukládána ve strukturované formě, splňovat logické podmínky kontextu jejich business užití, patřit do předem definovaného oboru hodnot pro každou datovou položku a být správné doménové hodnoty.

Novotný, Pour a Maryška (2012) systematizovali hlavní seznam příčin, kdy dochází k nečistotě dat, tyto příčiny zde budou rozebrány podrobněji společně s možnostmi jejich řešení.

Jedná se o:

1. **Chyby při manuálních vstupech dat, prohození číslic, překlepy, špatně zapsané kódy, hodnoty zapsané do nesprávného pole.**

V tomto případě je zásadní validace dat a používání číselníkových hodnot. Firemní aplikace přes jejich testování před nasazením do provozu velice často vykazují nedostatky ve validaci přípustných hodnot, které uživatel může vyplnit.

Pokud hodnota, kterou má uživatel v aplikaci vyplnit nebo se mu má naopak zobrazit, může pocházet jen z předem známé a omezené množiny hodnot, je systémové

tuto situaci vždy řešit vytvořením číselníku v produkční databázi, jeho hodnoty nabídnout k výběru nebo zobrazení v aplikaci a celý číselník pak propagovat dále do datového skladu. Toto by se mělo dít i v případě, že se jedná o číselník s minimem položek, který programátoři vždy rádi umisťují přímo v kódu aplikace. I když na straně datového skladu ve snaze o systémové řešení vznikne číselník na základě hodnot sdělených programátorem aplikace, je toto řešení zcela nesystémové. Při každé změně nebo doplnění číselníku to vyžaduje informování vývojáře DWH, na jeho straně to znamená provedení manuální úpravy, která z kapacitních důvodů nebo překážce v komunikaci nemusí být provedena okamžitě, takže v datovém skladu vzniknou oproti zdrojovému systému nekonzistence.

Samotná úprava číselníků přepisováním existujících hodnot na hodnoty nové je stav, který se dá řešit historizací číselníkové tabulky, takže zpětně je vždy dostupná hodnota, která odpovídala konkrétnímu primárnímu klíči v konkrétním okamžiku. Obecně se ale jedná o velice nebezpečnou praktiku, v případě potřeby nových položek číselníku je systémově správně zakládat nové položky a nepřepisovat původní, i když se již nepoužívají. Tímto se předejde zmatku v datech a také nebude třeba opravit nebo vyřadit velké množství již napsaného kódu ve strukturách databáze nebo SQL dotazech jednotlivých vývojářů nebo analytiků, kde se odkazuje na konkrétní hodnotu číselníku.

Data je třeba validovat průběžně, to platí zejména pro ta, která vkládá přímo uživatel a není možné je ošetřit číselníkem. Například při každém kontaktu se zákazníkem je užitečné ověřit základní identifikační údaje a kontakty, v případě potřeby pak transparentně a spolehlivě měnit původní data při zachování původních hodnot a jednoznačného identifikátoru zaměstnance, který změnu provedl.

2. Problém při konsolidaci dat z různých zdrojů

Zdrojem pro datové sklady jsou různé databáze a soubory, tyto zdroje často nejsou mezi sebou provázány. Nastává problém s vyhodnocením, které záznamy, případně atributy z různých záznamů ke stejnému objektu upřednostnit jako správné. Tento problém je řešen takzvanou unifikací, složitými procedurami, které dle sady pravidel tyto podmínky vyhodnocují. I tyto procesy však mají své limity a tak

vždy existuje určité procento záznamů, které díky nedostatku informací nebylo možné ztotožnit nebo byly ztotožněny chybně na základě mezery nebo chyby v pravidlech. Nepříjemný důsledek to mívá zejména v automatických kampaních, kdy jsou maily nebo sms zasílané neoprávněným osobám.

Řešením je navrhovat firemní aplikace promyšleně a robustně, s důrazem na jednoznačnou identifikaci objektu napříč všemi firemními procesy. Kvalitu záznamů z externích zdrojů často nelze ovlivnit, při konsolidaci může dojít ke ztotožnění na základě určitého atributu anebo k označení záznamu jako nespolehlivého. Účel využití dat určuje míru přísnosti pravidel, která jsou při vyhodnocování užívána.

3. Neoprávněné zkoušení a testování kódu a aplikací na ostrých datech

Vedení společností se z úsporných důvodů snaží minimalizovat množství serverů a licencí softwarových produktů. Pro potřeby vývoje by vždy měla existovat tři hardwarově a softwarově srovnatelná prostředí. První by mělo sloužit pro vývoj, druhé pro testování dokončených funkcionalit z vývojového prostředí a po úspěšném testování by mělo dojít k řízenému nasazení nové funkcionality na produkční prostředí, při tomto by vývojáři ani testeři neměli mít práva k přístupu na produkční prostředí.

Pokud existují pouze dvě prostředí s rozdílnými prostředky, přitom vývojové a současně testovací prostředí obsahuje zastaralá data nebo z nějakých důvodů není dostupné, pak skutečné otestování nové nebo upravené funkcionality proběhne v produkčním prostředí i se všemi možnými negativními následky. Jedním z nich může být například i poměrně obsáhlá změna ostrých dat, jejíž důsledky nebude možné napravit ani za použití nástrojů obnovy, jako jsou databázové zálohy nebo flashback transakce.

4. **Zásahy, většinou oprávněné, do zdrojových systémů, které byly učiněny bez návaznosti na následující komponenty systému.**

Takové situace nastávají v případě, když při vývoji nebo správě systému nejsou řešeny dopady jednotlivých změn nebo ve společnosti nejsou funkční procesy komunikace mezi jednotlivými odděleními vývoje a správy systémů. Jedná se vždy o systémový problém řízení procesů vývoje.

5. **Chyby při migracích a konverzích dat**

Migrace nebo slučování dat z různých databází jsou vždy rizikové procesy, při kterých někdy nedoje k přenosu nebo naopak dojde k duplikaci dat, navázání na jiný cizí klíč z číselníku, atd. Tyto procesy vyžadují velice zkušené správce a vývojáře databází, přesný a promyšlený plán postupu a velice přísné kontrolní fáze a mechanismy.

6. **Přímým přístupem do zdrojových databází v rámci sdílených nebo otevřených aplikací**

Vstup do zdrojové databáze by měl být řízený, práva přidělována smysluplně a systematicky a jakýkoliv externí zdroj dat by měl být podroben stejným validačním procesům, jako data vlastních systémů. Nová data by neměla být importována automaticky, ale teprve až po validaci a konsolidaci, ať už automatické nebo manuální.

7. **Chyby v návrhu databází, doménové nebo referenční integritě**

Jedná o klíčový faktor pro kvalitu dat v produkčních databázích, jenž se následně promítá do kvality dat i v datových skladech. Při rezignaci na základní databázové principy není možné očekávat korektní výstupy reportingu, případně nárůst pracnosti při eliminaci důsledků těchto pochybení je enormní a výrazným způsobem může omezit kapacity vývojářů datových skladů.

8. **Při konverzi datových typů**

Problémy nastávají zejména při implicitních konverzích, kdy převod z jednoho datového typu na druhý je ponechán na aplikaci nebo databázi. Převod na jiný datový typ může být po formální stránce úspěšný, ale data pak jsou v cílové databázi nebo souboru v nesmyslném tvaru. Problém může způsobit i rozdílné jazykové prostředí, vzhledem k odlišnému chápání desetinné čárky a tečky dojde při konverzi k chybě nebo k posunu čísla do jiného řádu.

Řešením je vždy explicitně konvertovat každý atribut, výslovně uvést cílový datový typ a případně masku. V případě načítání externího souboru nástrojem ETL je jedním z osvědčených postupů načít všechna data jako text a v dalším kroku je řízeně převést na požadovaný datový typ. Problematické záznamy se při převodu vyřadí a následně se pro ně vytvoří nové pravidlo nebo se reklamují u poskytovatele externího zdroje.

3.2.3. Centralizované řízení datových toků a jejich prezentace

Jazyk SQL má velice jednoduchou syntaxi, jejíž osvojení je poměrně snadné. V podnicích často výstupy z datového skladu nezískávají pouze specialisté SQL, ale i pracovníci jiných specializací, například business analytici, pracovníci marketingu, specialisté na statistické výstupy.

Přínosem tohoto přístupu jsou rychle dostupné analýzy oprávněným pracovníkům bez požadavků na kapacity specialistů IT. Negativem je časté porušení principu „jediné pravdy“ v podnikových datech a spotřeba prostředků databáze na úkor jiných uživatelů nebo procesů. Úskalí jazyka SQL je optimalizace, jednoho a toho samého výsledku je možné dosáhnout různě napsaným SQL dotazem, ale optimalizovaný kód je vykonán v řádu několika vteřin až minut, zatím co vykonání neoptimalizovaného kódu může trvat i mnoho hodin.

Podle (DYCHÉ, LEVY, 2006) "demokratizace" databází přispěla k šíření redundantním a vzájemně si odporující dat napříč společností. Šíření duplikátu, chyby, nesynchronizovaných dat nebo chybějících informací stav zhoršuje.

Řešení této situace je systémové, pokud mají mít přístup do datového skladu i jiní pracovníci, měli by být vyškoleni v psaní optimalizovaného kódu SQL. Jejich dotazy by měly být schváleny specialisty a v případě opakovaného využívání by z nich měly být vytvořeny standardní reporty v některém z nástrojů pro reporting. Dalším řešením je vytvořit datovou základnu v některém z enterprise nástrojů, kdy pak již nebude nutné spouštět dotazy přímo v databázi, ale tvořit jednotlivé reporty kombinací souvisejících entit prezentační vrstvy reportingového nástroje.

3.2.4. Minimalizace požadavků, maximalizace užitku

V provozním či ad-hoc reportu je snadné přidat nebo odebrat sloupce a při příštím generování bude výsledková sada obsahovat požadovaná data. Pokud je však požadavek na změnu datového tržiště v datovém skladu přidáním nového klíčového atributu, zpravidla to znamená úpravu několika vrstev. Dále to vzhledem ke změně granularity může znamenat i možnou ztrátu dosavadních dat nebo přijetí skutečnosti, že některé hodnoty datového tržiště jsou na rozdíl od jiných dostupné až od určitého data.

Požadavky na rozšíření reportu nebo datového tržiště jsou častým důsledkem vývoje potřeb uživatelů. Pokud jsou ale požadavky na zásadní změny datových tržišť nepřetržité, svědčí to o neexistenci jasné a kontinuální koncepce firemního reportingu nebo o nízké kvalitě analýz před započetím vývoje.

Důležité je též zvolení architektury dimenzí a faktových tabulek již v datovém skladu nebo na datové vrstvě enterprise nástroje. Efektivní je implementovat pouze ty atributy, pro které již v době analýzy existuje jasné business využití a případně i klíčové atributy nezbytné pro další rozšíření. Nadbytečné agregované sloupce faktových tabulek pouze zaplňují místo úložiště a zhoršují odezvu jednotlivých

reportů. Stejně tak vazby více objemných faktových tabulek stejné granularity přes shodné dimenzionální tabulky mívají výkonnostní problémy. Vhodnějším řešením je více specializovaných datových tržišť s některými redundantními atributy, pokud není možné všechna potřebná data z logických nebo výkonnostních důvodů umístit do jedné faktové tabulky.

Podle YEOHA (2010) se vzhledem ke složitosti a délce implementace může BI projekt dostat zcela mimo kontrolu. Iterativní vývoj zajišťuje, aby se skutečné úkoly projektů neztratily v chaotických projektových prostředích. Řízený a opakující se vývoj pozitivně podporuje úspěch implementace BI systému.

3.2.5. Důvěra v data, vývojový tým, celofiremní využívání reportingového systému

Lamberge(2012) popisuje situaci při jednom z projektů, kdy se během vývoje datového skladu pracovních schůzek zúčastňoval jeden z nejvýše postavených manažerů ve firmě, který disponoval všeobecnou přirozenou autoritou. Svou přítomností demonstroval důležitost projektu a zvýšil tím zapálení zainteresovaných zaměstnanců v období vývoje skladu i ochotu přijetí po jeho nasazení.

Tento faktor je velice důležitý. Bez pocitu každého zaměstnance, že se jedná i o jeho datový sklad nebo reportingový nástroj a proto je nutný i jeho proaktivní přístup při analýzách nebo validaci dat, bude vždy dodáno řešení, jehož využití bude limitováno svým užitkem nebo přijetím.

Uživatelé reportu musí chápat význam testování a pilotního provozu, ve kterém se odstraňují funkční nebo logické chyby u nově vytvořených reportů. Automatický firemní reporting by měl být co nejvíce spolehlivý, data v obvyklou dobu by měla být na obvyklých místech a zobrazované hodnoty správné. Důvěra ve zdroj dat se totiž buduje dlouhodobě, ale ztrácí se velice rychle.

Řešením je tedy podpora vedení, budování otevřeného přístupu k novým postupům a nástrojům, podporování proaktivního přístupu při vývoji nebo užívání

a vytvoření natolik výkonného vývojového, testovacího a produkčního prostředí, aby tím nebyly degradovány možnosti jednotlivých částí BI.

3.3. Fáze vývoje reportu

Podle Novotného, Poura a Slánského (2013) existují tři způsoby tvorby celkových řešení, prvním jsou nezávislá datová tržiště definovaná Ralphem Kimballem v osmdesátých letech minulého století. Druhým způsobem je celkové řešení konsolidovaného datového skladu a tím třetím přírůstkové řešení konsolidovaného datového skladu.

Každá firma má individuální potřeby a priority, proto pro ni může být výhodná jiná volba, jako nejefektivnější se však jeví přírůstkové řešení. Nejprve se vytvoří celková koncepce BI řešení, která obsahuje souhrn všech uživatelských požadavků, jejich priority a hrubý časový harmonogram. V celkové koncepci jsou již definovány jednotlivé přírůstky, jejich návaznost, nákladnost a návratnost investic. Poté dle harmonogramu započne implementace konkrétních přírůstků.

Výhody tohoto přístupu jsou rychle dodávané přírůstky, jejichž implementace se snadněji kontroluje z hlediska kvality i nákladů. Navíc v průběhu jednotlivých přírůstků dochází k procesu učení jak na straně zadavatele, tak vývojářů. Zadavatel postupně s jednotlivými přírůstky získává větší znalosti o možnostech BI řešení, i přes specifikaci v celkové koncepci si mnohem více uvědomuje důležitost nebo zbytečnost konkrétních přírůstků nebo jejich atributů a je lépe schopen komunikovat s vývojáři a upřesňovat své požadavky. Vývojáři naopak lépe pochopí způsob užívání jednotlivých přírůstků BI zadavatelem a v následných přírůstcích mohou být zvolena optimálnější datová nebo vizuální řešení, aniž by to zvýšilo pracnost při tvorbě přírůstku.

Nevýhodou přírůstkového přístupu je časová náročnost při vypracování celkové koncepce, proto dodávka prvních přírůstků je opožděna oproti předchozím dvěma přístupům.

V této práci budou popsány fáze vývoje reportu jako přírůstkového řešení.

3.3.1. Rozhodnutí o vývoji reportu

3.3.1.1. Rozpoznání oblasti vhodné pro pokrytí reportem

Oblasti a procesy vhodné pro pokrytí reportingem by měly vycházet z reálných potřeb zaměstnanců. Proto by vedení firmy mělo vést zaměstnance k proaktivnímu jednání, aby sami iniciovali tvorbu nových reportů v případě, že je pro svoji činnost vyhodnotí jako důležité a nezbytné. V ideálním případě existuje podnikový formulář, který při zadání požadavku zadavatele navádí k vyplnění všech důležitých informací.

Tyto nové požadavky zaměstnanců by měly projít schvalovacím procesem business analytiků, při kterém by byla ověřena oprávněnost, unikátnost a formální správnost požadavku.

Zdaleka ne všechna data požadovaná zaměstnanci v datovém skladu a následně reportech jsou užitečná. Gemignani (2015) při vývoji BI dělí data na důležitá, jejichž změna v reportu vyvolá konkrétní akci příjemce a na data zbytečná, odvádějící pozornost, jejichž změna nevyvolá žádnou reakci příjemce.

3.3.1.2. Revize existujících reportů

Pokud se jedná o začínající firmu, kdy se vytváří se kompletně nový reporting, odpadá potřeba kontroly stávajících reportů.

V opačném případě je nutné tuto kontrolu provést, aby nebyl požadován vývoj reportu, který již existuje nebo aby se požadovaná data získala pouze drobnou úpravou jiného reportu. Tato kontrola může být snadná, pokud existuje aktualizovaný seznam existujících reportů i se základním popisem, jakou problematiku konkrétní report řeší. V případě, že žádný z existujících reportů neřeší požadovanou problematiku, mělo by

následovat vypracování strukturovaného zadání požadavku a v případě jeho schválení i všechny ostatní fáze vývoje reportu.

Vytvoření specifikace reportu

Cílem specifikace požadavku je vytvořit konkrétní zadání pro rozhodnutí o vývoji reportu a následné fáze vývoje. Specifikace by měla obsahovat:

- Vyhodnocení aktuálního stavu BI
- Definování přírůstku
- Určení harmonogramu, nákladové stránky a ekonomiky přírůstku

Aby bylo možné definovat harmonogram a náklady spojené s vývojem, musí proběhnout hrubá analýza požadavku, kdy se nezvažují ty největší detaily řešení, ale náročnost vývoje požadovaných změn se odhaduje vzhledem ke zkušenostem z vývoje z předchozích přírůstků.

Specifikace po schválení vývoje přírůstku bude využita jako vstup do dalších fází vývoje.

3.3.1.3. Rozhodnutí o vývoji

Pokud byla revize existujících reportů negativní, nastává fáze rozhodnutí, kdy a kým bude požadovaný report vyvinut.

Do určení termínu vývoje se většinou promítá důležitost požadovaného reportu, aktuální vytíženost interních či alokace externích vývojářů, existence schvalovacích a vývojových cyklů ve společnosti, schválení investic mateřskou společností. Po zvážení uvedených skutečností budou termíny promítnuty do harmonogramu ve specifikaci požadavku.

Do rozhodnutí o osobě vývojáře se promítají faktory, zda se jedná o standardní report v běžně používaných nástrojích ve společnosti, zda společnost disponuje

vlastními vývojáři, kteří jsou schopní takový požadavek zvládnout a mají potřebné kapacity. V takovém případě je logickým rozhodnutím interní vývoj, který je levnější, v případně volných kapacit rychlejší a vzhledem ke znalosti vlastních procesů a podnikové kultury se bude nejvíce blížit ke shodě s požadovaným zadáním. V opačném případě report vytvoří externí firma, náklady na externí vývojáře bývají zpravidla více jak dvojnásobné, ale v tomto případě se nejedná o mzdové náklady.

3.3.1.4. Rizika spojená s rozhodnutím o vývoji

Rizika v této fázi spojená s činností vývojářů reportů se týkají zejména hrubého odhadu pracnosti užitého pro specifikaci, který se může značně lišit od skutečné pracnosti, která je známa až v době dokončení. Proto by v předpokládané pracnosti měla být vždy určitá rezerva pro překážky ve vývoji, které v době odhadu nebylo možné předpokládat.

Neexistence nebo nedbalá evidence existujících reportů má za následek téměř vždy nadbytečné náklady na vývoj.

3.3.2. Analýza reportu

Cílem analýzy je prověřit požadavek na vývoj, zhodnotit kvalitu a dostupnost existujících datových zdrojů a navrhnout řešení pro splnění požadavku. Závěrem analýzy může být i zjištění, že pro úspěšný vývoj reportu nejsou splněny všechny podmínky, například chybí nějaký důležitý zdroj nebo některý proces v produkčním systému není zaznamenáván takovým tak, aby mohl být reportován požadovaným způsobem.

3.3.2.1. Revize požadavku a zpřesnění

I přes kvalitně provedenou specifikaci je nezbytné znovu prověřit požadavek se zadavatelem reportu. Je běžné, že od zadání dojde k posunu, a tak je třeba ověřit, zda je zadání platné tak, jak bylo uvedeno ve specifikaci, či je třeba doplnit nebo pozměnit nějaký atribut.

Při ověřování požadavku je třeba ověřit:

- Cíl reportu
- Seznam atributů a jejich zdrojů
- Způsob přijímání reportu – online, mailem, sms
- Typ reportu – ad-hoc, provozní strategický
- Periodu reportu – hodinový, denní, týdenní, měsíční, roční
- Seznam uživatelů reportu
- Čas generování reportu – první pracovní den v týdnu, měsíci

3.3.2.2. Detailní analýza

V této fázi se pověřuje realizovatelnost požadavku z hlediska finančního, technologického, datového a personálního. Součástí této fáze je ověření kvality a dostupnosti vstupních dat. Zjišťuje se, zda jsou data dostupná v primárním systému v potřebné formě a kvalitě.

Výsledkem analýzy by mělo být konstatování, že na základě všech těchto hledisek je možné požadavek splnit nebo vzhledem k některému hledisku je za současné situace požadavek nerealizovatelný. Jednou z možností je i nový požadavek na úpravu produkční databáze, neboť je to nezbytné pro dokončení vývoje požadovaného reportu.

V případě, že požadavek je realizovatelný, ve fázi analýzy probíhá rozhodnutí, zda je možné požadovaný report vyvinout úpravou některého existujícího nebo je nutné vytvořit kompletně nový report.

Součástí analýzy je i zvolení nástroje reportingu, ve kterém bude report zobrazen. Velké firmy disponují více nástroji, rozhodnutí může být ovlivněno vhodností uvedeného nástroje nebo skupinou uživatelů, kteří budou report přijímat.

Výstupem analýzy by mělo být doplněné nebo upravené zadání požadavku, které bude podkladem pro ostatní fáze vývoje.

3.3.2.3. Rizika spojená s fází analýzy

I přesto, že ve fázi analýzy dochází ke zpřesnění požadavku a zadavatel může oproti původnímu zadání požadovat úpravu nebo doplnění více atributů, upravený požadavek by stále měl vycházet z požadavku původního. Pokud by při zpřesnění požadavku bylo požadováno zcela jiné zadání, je na místě požadavek vrátit zpět do fáze rozhodnutí o vývoji reportu, neboť rozhodnutí bylo učiněno pro zcela jiné zadání a náklady vývoje včetně potřeby alokace vývojářů se od původního zadání může značně lišit.

Rizikem je též podcenění nebo nezahrnutí některého aspektu do analýzy. Následkem může být navýšení pracnosti až zbytečnost celého vývoje, pokud se na problém přijde až při vývoji nebo po nasazení reportu.

3.3.3 Návrh reportu

Cílem fáze návrhu reportu je navrhnout datové modely datového skladu, případně datových tržišť a dalších databázových komponent a s tím spojené aspekty uložení, nárůstu a práv. Dále jsou předmětem návrhu transformace dat ETL a struktury BI nástroje, ve kterém bude report vytvořen.

3.3.3.1. Modelování a návrh BI řešení

Podle Poura (2012) dimenzionální modelování zahrnuje:

- vymezení všech dimenzí, jejich obsahu a hierarchie, které budou u reportu použity
- vymezení faktů, které budou sledovány a jejich organizace do jednotlivých struktur
- specifikace vazeb mezi dimenzemi a fakty

Faktové tabulky

Zásadní vlastností u tabulek faktů je jejich granularita, data je v konečných strukturách možné z důvodu výkonnosti agregovat na denní, týdenní nebo měsíční úroveň podle klíčů jednotlivých dimenzí, datový sklad by ale měl disponovat takovou kapacitou, aby umožnil uložení zdrojových dat v jejich původní granularitě. Toto je vždy důležité při ověřování výsledků a stejně tak i pro tvorbu nových reportů nad stejnými faktovými daty, které ale mají mít jinou granularitu. V případě, že se do faktové tabulky dostanou data z různých zdrojů, měla by být zajištěna shodná granularita dat.

Další aspekt, který by měl být řešen u faktových tabulek, jsou měrné jednotky. Měrná jednotky by se měla objevit přímo v názvu konkrétního atributu, jinak by ale v datovém skladu vždy vše mělo být převáděno na stejnou míru, pokud

jde například o finanční částku, měla by být přepočítána na kurz CZK k datu, kdy položka vznikl a. Tato filosofie by měla být plošně užívána v celém datovém skladu.

Aplikace agregace hodnot ve faktových tabulkách v datovém skladu by měla být nezávislá na reportingovém nástroji, vzhledem k možnostem současných manažerských a reportingových nástrojů by hodnoty měly být agregovány vždy jen na počty a sumy, všechny ostatní agregace, jako např. průměry, minima, maxima, percentily by měly být ponechány na reportingovém nástroji. V případě hierarchií a drill-down by nebyl počítán například průměr, ale průměr z průměru a tam se již výsledná data mohou dost podstatně lišit.

Dimenzionální tabulky

Tabulky dimenzí vycházejí z číselníků v produkčním systému, většinou se do datového skladu propagují 1:1, jen se odstraní nepotřebné sloupce. Podle Poura, Maryšky a Novotného (2012) by k jedné faktové tabulce mělo být napojeno maximálně 15 dimenzí. Pokud by při analýze vznikla potřeba pro více jak 15 dimenzí, je efektivní některé dimenze sloučit a vytvořit v nich hierarchie. Opět by zde mělo zvítězit pravidlo skutečné potřeby - dimenze, které sice logicky k datům náleží, ale nikdy, ani výhledově nebudou použity, by v případě velkého počtu dimenzí měly být vyřazeny hned na počátku.

Architektura datového skladu a datových tržišť

Existují dva přístupy spojování dimenzionálních a faktových tabulek:

- sněhová vločka – spojování tabulek v normalizované formě je podobné principu spojování tabulek v produkční databázi, přes určité výhody je toto řešení nevýkonné a z tohoto hlediska se z pohledu vývojářů datových skladů nejedná o dobrou praktiku

- hvězda – jedna faktová tabulka je obklopena několika dimenzionálními tabulkami, toto řešení je v současné době v datových skladech

preferované. Jelikož cena úložišť dat oproti minulosti již není tak nákladnou položkou v řešení BI, i v případě, že by dvě faktové tabulky bylo možné spojit logicky přes stejné dimenzionální klíče, toto řešení se z důvodu výkonnosti nedoporučuje a část dat se raději replikuje. Smysluplná redundance dat v datovém skladě totiž není prohřeškem proti pravidlům tvorby datového modelu jako v produkční databázi.

Napojení faktových a dimenzionálních tabulek

Spojení faktové a dimenzionální tabulky se děje za pomoci cizích klíčů, kdy ve faktové tabulce se jeden a více záznamů odkazuje právě na jeden záznam dimenzionální tabulky, přičemž každý záznam v tabulce faktů by svou kombinací cizích klíčů měl být jedinečný.

V případě, že by se do datového skladu propagovala data pouze z jediné databáze nebo více databází se společnými, globálními číselníky, by bylo možné všechny číselníky nahrát jako dimenze a do faktových tabulek nahrát se záznamy i cizí klíče, protože vazba by stále odpovídala skutečnosti.

V realitě však většinou existuje mnoho databází s vlastními číselníky, které nejsou provázané a i v případě stejného číselníku můžou být pod stejnými primárními klíči jiné hodnoty. Proto se při vyhledávání klíčů využívají převodníkové tabulky, kdy více hodnot z navzájem různých zdrojů může vést ke stejnému záznamu v dimenzionální tabulce. Převodníkové tabulky je však nutné manuálně udržovat, což je pracné a nespolehlivé.

Druhým způsobem je vyhledává na základě unikátní hodnoty, kdy klíčem je několik spojených hodnot, definujících mimo jiné i zdrojový systém, odkud data pochází a na základě hodnoty se vyhledá správný klíč dimenzionální tabulky.

Postupy dimenzionálního modelování

Podle standardní metodiky Kimballa(2006) se dimenzionální modelování dělí na 5 částí:

- Příprava řešení dimenzionálního modelu
  - Zhodnocení závěrů úvodní studie
  - Specifikace priorit podnikových procesů směrem k BI
  - Rekapitulace požadavků
  - Vyhodnocení stavu v databázi a aplikaci IS

- Návrh hrubého dimenzionálního modelu
  - Základní specifikace dimenzí, charakteristik, atributů
  - Návrh sledovaných ukazatelů, KPI, charakteristik a vztahů k dimenzím

- Analýza a návrh datového skladu a datových tržišť
  - Návrh datového modelu datového skladu nebo datového tržiště
  - Analýza dostupnosti dat
  - Detailní návrh charakteristik atributů dimenzionálních a faktových tabulek

- Verifikace návrhu dimenzionálního modelu
  - Verifikace detailního návrhu dimenzionálního modelu či prototypu uživateli
  - Zpracování protokolu z průběhu oponentur
  - Specifikace problémů a zadání úprav

- Finální kompletace a dokumentace dimenzionálního modelu
  - Kompletace celkové dokumentace dimenzionálního modelu
  - Zadání pro implementaci řešení

Návrh vrstev řešení, modelování datového skladu a tržišť

Jedná se o konkrétní návrh plnění jednotlivých vrstev datového skladu a datových tržišť včetně návrhu logického datového modelu jednotlivých struktur.

Návrh reportů včetně analytických pravidel

Určení obsahu a formátu požadovaných reportů včetně určení limitních hodnot ukazatelů, které mohou být pro uživatele použity jako varování, problém nebo příležitost.

Návrh a realizace prototypů

Na vzorku dat se vytvoří prototyp řešení, které validuje zadavatel reportu. Případné připomínky se promítnou do úprav výsledného řešení.

3.3.4 Návrh technologické platformy přírůstku

V této části návrhu se definují softwarové a hardwarové nároky spojené s přírůstkem řešení. Na základě dimenzionálního modelování vznikne fyzický datový model datového skladu a datových tržišť s uložením dat s ohledem na požadovanou logiku a výkonnost řešení BI. Dále se řeší předpokládaný nárůst objemu dat v příštích letech, přístupová práva, nároky na zajištění provozu a bezpečnosti.

Nárůst objemu dat v datovém skladu by se však měl kontrolovat standardními mechanismy průběžně. Zkušený vývojář by měl znát přibližný odhad nárůstu dat řešeného přírůstku již ve fázi analýzy a v případě nadměrně objemného řešení by se vývoj neměl posunout do dalších fází bez adekvátního navýšení kapacit úložiště datového skladu.

Zásadním aspektem je historizace vstupních dat. V případě systému s mnoha změnami nemusí rapidně narůstat počet záznamů v tabulkách, ale změny již existujících záznamů mohou znamenat značný nárůst dat v historických tabulkách. Z tohoto důvodu by v tabulkách datového skladu neměly být žádné zbytečné atributy, které nemají jasné využití v následném reportingu nebo ad-hoc dotazech.

3.3.5 Návrh transformací dat

V této části návrhu se řeší pravidla pro transformaci a kvalita dat. Specifikuje se, jak budou data transformována na různých vrstvách, jak budou vyřešeny chybějící záznamy nebo záznamy s hodnotami NULL. Dále se řeší chybné záznamy, které se do zdrojových dat dostaly díky chybám v databázovém návrhu produkčních databází nebo chybám uživatelů. Součástí je i řešení integritních problémů, tedy business vazeb, které sice logicky existují a v datovém skladu jsou požadovány, ale v produkční databázi neexistují.

3.3.6 Rizika spojená s modelováním a návrhem BI řešení

Rizik v této fázi vývoje je mnoho zásadní je dimenzionální modelování, kdy návrh jednotlivých dimenzionální a faktových tabulek a jejich spojení by neměl být optimální jen z hlediska business potřeb, ale také z hlediska výkonnostního. Nejedná se však jen o výkonnost v době uvedení řešení do provozu, ale hlavně o výkonnost v následných letech užívání, kdy počet záznamů ve faktových tabulkách poroste i řádově.

Dalším rizikem je podcenění počtu vrstev v BI řešení, například v případě úsporného řešení k úložišti, kdy nejsou dostupná zdrojová data ve své původní granularitě. Během užívání reportu vždy po určité době dojde k požadavku na ověření zobrazovaných dat, protože jiné zdroje informací vzhledem k odlišnému způsobu nebo době generování vykazují jiné hodnoty. Pokud není v takové chvíli možné předložit zdrojové záznamy, report se stává nedůvěryhodným až do doby, kdy bude nalezena a zveřejněna příčina rozdílu.

Při návrhu způsobu ETL transformací proti sobě vždy stojí možnosti samotných databází, jako jsou databázové linky a uložené procedury, proti možnosti aplikací specializovaných na ETL procesy. Základními hledisky jsou funkčnost, transparentnost, výkonnost a spolehlivost. U objemných dat, kdy se dohledávají klíče dimenzí, je to vždy o zvážení možnosti optimalizovaných databázových dotazů oproti použití tzv. lookupů v ETL nástrojích, které pro každý záznam procházející tokem dat vyhledávají klíče z dimenzionálních tabulek.

V uvedené fázi poměrně často dochází k navýšení objemu požadavku a tedy i pracnosti. V tomto směru by vždy měla existovat vazba zpět k rozhodující autoritě ohledně navýšení nákladů a možnému ohrožení termínu. Teprve po akceptaci tohoto navýšení by měl vývoj pokračovat do další fáze.

3.3.7 Implementace reportu

3.3.7.1. Implementace

V této fázi se vyvíjí jednotlivé části BI řešení podle návrhu a zadání přírůstku. Pokud byly fáze zadání požadavku a analýzy provedeny pečlivě, neměly by v této fázi nastat nějaká překvapení, která by ohrozila úspěšné dokončení nebo nějakým zásadním způsobem navýšila pracnost.

Postupně se implementují databázové struktury na jednotlivých vrstvách, nastavují se ETL procesy, validují se počty přenesených záznamů co do jejich počtu a shodných hodnot se zdrojem dat. V případě implementace ETL procedur by měla být řešena optimalizace kódu SQL, k tomuto slouží exekuční plány ve vývojových nebo monitorovacích nástrojích databáze. Podle řešení se implementují OLAP kostky, řešení data miningu, případně jednotlivé vrstvy manažerských nástrojů.

Při spojování faktových tabulek nebo datových tržišť datového skladu s dimenzionálními tabulkami na fyzické vrstvě manažerských nástrojů je z výkonnostních důvodů doporučeno využívat schéma hvězdy stejně, jako v datovém skladu. Nástroje většinou umožňují spojení více faktových tabulek přes společné dimenzionální tabulky. Pokud ve faktových tabulkách bude stejná granularita, pak výsledek poskytne i smysluplná data, z výkonnostního hlediska u objemných tabulek a dimenzí to může znamenat problém. Standardním řešením je pak zadat nový požadavek na úpravu některé z faktových tabulek a dodat tam chybějící atributy i za cenu redundance dat. Jedním z hlavních úkolů manažerských nástrojů je možnost prezentace aktuálních dat on-line a tak by výstup z těchto nástrojů měl být v řádu pár vteřin. Pokud by tomu tak nebylo, nástroje by ztratily svou funkci on-line prezentace jediné aktuální pravdy a návrat k souborům na přenosných médiích nebo sdílených úložištích.

Poslední činností je vizualizace reportu, kdy jednotlivé prvky reportu a jejich ergonomie by měly odpovídat zadání reportu. Zásadní je samotné umístění reportu na panelech nebo stránkách manažerského nástroje, toto by se mělo dít na základě shodné

business oblasti nebo na základě personalizace reportů pro konkrétní skupiny uživatelů.

Vizualizace by měla zajistit, aby report byl pro uživatele přívětivý, názvy sloupců byly pojmenovány jednotně a jednalo se o jednotlivé pojmy firemního business slovníku. Hodnoty by měly být formátovány pro lepší čitelnost, sumy zaokrouhlovány na celá čísla a oddělovány po tisících, aby byl na první pohled zřejmý rozdíl a řád hodnot. Výjimkou jsou procenta, kde jsou důležitá i desetinná čísla. Některé nástroje umožňují i zobrazení oznámení, že pro zadané parametry neexistují žádná data.

I přes popis nebo návrh zobrazení v zadání požadavku až teprve při předložení prvního výstupu uživatel vysloví svou spokojenost nebo připomínky k ovládání reportu. Pokud byly zadavateli známy možnosti nástroje reportingu, pak tyto změny jsou většinou kosmetické a neznamenají navýšení pracnosti. Změny vizualizace v enterprise nástrojích není třeba vývojářů, ale zvládne je i vyškolený a zkušený business analytik, který ale z bezpečnostních důvodů disponuje právy pouze na úpravu prezentační vrstvy.

3.3.7.2. Ověřování při implementaci reportu

Ve fázi analýzy byly vytvořeny vzorky dat nebo prototypy řešení, tyto vzorky a prototypy však neobsahovaly všechna data. V případě postupné implementace jednotlivých vrstev je osvědčenou praktikou validace dat ze strany uživatelů, kteří s daty pracují. Tyto validace by měly zajistit úplnost dat, kontrolu začlenění dat do správných položek jednotlivých dimenzí.

Teoreticky již by neměla nastat situace, aby v této fázi zadavatel navrhoval nové úpravy reportu, které by znamenaly dodatečný nebo kompletně nový vývoj databázových struktur a ETL procesů. V praxi se tak občas stává v případě, že firma prochází dynamickými změnami, neexistuje jednotné a efektivní projektové řízení,

požadavky nejsou spojeny se skutečnými potřebami firmy, ale s konkrétními požadavky jedinců, s jejichž odchodem nebo přechodem na jiné oddělení končí svůj životní cyklus. Pokud tato hraniční situace nastane, je na zvážení rozhodovacích autorit, zda vývoj bude dokončen dle zadání a specifikace a následně bude podán změnový požadavek nebo bude vývoj zcela zastaven.

3.3.7.3. Rizika při fázi implementace

Rizikem fáze implementace je vývoj odlišný od požadovaného zadání a specifikace. Vývojář by měl být zkušený, aby řešení bylo spolehlivé a optimalizované. V některých článcích o kvalitě vyvinutého software je například uvedeno, že by ve vývojovém týmu neměli být žádní junior programátoři. Toto nebezpečí hrozí zejména u externího dodavatele, kdy se na různých projektech zaučují nezkušení vývojáři s minimálním dohledem, kvalita jejich práce nemůže odpovídat kvalitě práce vývojáře s mnohaletou zkušeností. Součástí smlouvy o provedení přírůstku by tedy mělo být i vyhrazené právo zadavatele prověřit, zda schopnosti vývojáře externího dodavatele odpovídají požadavkům.

Rizikem je započít fázi vývoje reportu v momentě, kdy ještě nejsou ustáleny struktury nebo procedury v produkčních databázích. Toto se stává u celofiremních release, kde vývoj jednotlivých oddělení na sobě závisí. Vývoj v datovém skladě následuje vždy až po vývoji v primárních databázích a měl by tedy začít teprve v momentě, kdy řešení v primární databázi bylo nasazeno do provozu a je stabilní. Ideálně by vývoj datového skladu a reportu měl začít až v následném release. Jinak hrozí, že produkční řešení bude nasazeno až na konci release, vstupní struktury a data budou dostupná až těsně před koncem termínu, takže vývoj části BI řešení v datovém skladu a manažerských nebo reportingových nástrojích bude probíhat zrychleně, což sníží to kvalitu implementace. K této situaci dojde i v případě, že vývojáři produkčního prostředí předem dodají návrh struktur a dat a vývojáři datového skladu začnou s implementací s předpokladem, že vstupy budou odpovídat návrhu. Při implementaci v primárních systémech může dojít ke změnám, což by u hotového řešení v datovém skladu znamenalo změny nebo

přizpůsobování řešení. To opět může mít negativní dopad na spolehlivost či výkonnosti řešení v datovém skladu.

3.3.8 Testování reportu

Ve fázi testování se ověřuje funkčnost a spolehlivost plnění jednotlivých vrstev datového skladu a datových tržišť, případně plnění fyzické vrstvy manažerských nástrojů, prezentace dat v reportu, funkčnost jednotlivých uživatelských prvků reportu a zasílání reportu cílovým skupinám uživatelů.

3.3.8.1. Testovací prostředí

Ideálním stavem je existence tří prostředí: vývojového, testovacího a produkčního, přičemž tato prostředí disponují stejným hardware, software potřebným k vývoji a databáze obsahují shodná data.

Na vývojovém prostředí se vyvíjí nové struktury a procedury, zde mohou být jednotlivé objekty v rozpracovaném a nevalidním stavu.

Po ukončení vývoje se všechny části řešení nasadí na testovací prostředí, kde jsou přístupné testerům, ale skryté před koncovými uživateli. Zde by se měly provést všechny datové, aplikační a výkonnostní testy. Mělo by být ověřeno, že nové nebo změněné databázové struktury, objekty a procedury jsou ve validním stavu a nezpůsobily, že jiné původní databázové struktury, objekty a procedury jsou nevalidní. Otestována by měly být i výkonnost ETL procesů a případně procedur, které plní jednotlivé vrstvy nebo konečná datová tržiště.

V praxi bohužel bývá realita taková, že firmy obvykle hodlají investovat pouze do dvou prostředí, a tak jedno plní roli vývojového a testovacího prostředí a druhé produkčního prostředí. Vývojové a současně testovací prostředí pak často nemá stejný objem dat, neobsahuje všechny hodnoty číselníků a tak skutečný test výkonnosti a spolehlivosti skutečně proběhne až po nasazení do produkčního systému. Potom zcela reálně hrozí, že nenastane chyba v důsledku hodnoty, která na testovacím prostředí v datech nebyla nebo tvorba reportu trvá nepřiměřeně dlouho, neboť optimalizace byla provedena na zcela jiném objemu dat.

Takový případ se dá řešit použitím databázových linků, kdy se na vývojovém a testovacím prostředí do tabulek shodných struktur importují data z produkčního prostředí. Potom je možné ověřit chování ETL procesů a vytvoření reportu na reálných datech. Jedná se však o zcela nesystémové řešení, neboť při nasazení nemusí být všechny tyto linky odstraněny a v produkčním systému to pak může způsobit pád nebo výkonnostní problém.

Systémové řešení tedy je udržovat tři prostředí ve shodném stavu, toto však vyžaduje vyšší náklady a částečnou alokaci kapacit správců systémů.

3.3.8.2. Osobnost a schopnosti testera

Testování nového reportu nemůže provést stejný vývojář, který report vytvořil. Pokud firma tvoří komerční software, u něj se předpokládá vysoká kvalita a tomu pak také odpovídají ceny licencí, pak schopnosti testerů, kteří software vyvíjí, musí odpovídat schopnostem programátorů, kteří projekt vyvinuli. Samozřejmě i zde existují hierarchie specialistů, kdy na vedoucích programátorech a zkušenějších programátorech týmu závisí hlavní odpovědnost a méně zkušení vývojáři pod jejich vedením a kontrolou testují menší části.

Pro účel kontroly reportu je třeba, aby tester byl schopen ověřit správnost dat, správnost chování prvků aplikace, která report zobrazuje a správnou distribuci reportu. Při své činnosti by tester měl znát obsah zadání požadavku a specifikace po fázi analýzy pro vývoj reportu. K výsledkům práce vývojáře by měl přistupovat kriticky,

nepřejímat předpoklady jako jistoty a při testování aplikace být kreativní v testování nestandardních přístupů uživatelů k aplikaci.

Tímto způsobem tester může odhalit chyby v datech nebo nastavení, či chování aplikace a přispět tak tomu, že budou opraveny ještě před nasazením k cílovým uživatelům, u kterých by nesprávnost dat nebo nefunkčnost aplikace a zprovoznění do požadovaného stavu až na několikátý pokus mohlo vyústit až v nedůvěru v reporting nebo konkrétní report, hledání náhradních zdrojů informací a nakonec třeba i k odmítnutí vyvinutého reportu jako zdroje opravdivých informací.

3.3.9 Validace dat

V momentě existence výstupu reportu je dobrou praktikou ověřit správnost dat, ať již například proti původnímu ad-hoc reportu založeném pouze na SQL kódu nebo na vytipovaných záznamech prostřednictvím firemní aplikace pracovníky, kteří jsou schopni data validovat.

Přestože data v jednotlivých databázových vrstvách řešení a datovém tržišti mohou být správná a úplná, zobrazení v nástroji reportingu nebo manažerském nástroji může vzhledem k aktivitám při tvorbě reportu nebo vlastnostem aplikace či zvoleného prohlížeče ukazovat data nesprávná, neúplná, chybně agregovaná.

3.3.10 Testy uživatelského rozhraní

Před předáním reportu k užívání je nezbytné otestovat všechny ovládací prvky zobrazení, jako jsou filtry, hierarchie, návraty na původní obrazovky, linky, správnost a celistvost názvů a popisků. Špatné nastavení těchto prvků může způsobit neúplné nebo mnohonásobné vracení hodnot i z datově správného a validovaného zdroje.

Opakované testy uživatelského rozhraní mohou souviset i s pilotním provozem, například u reportů v Oracle BI, které obsahují data za několik minulých

období, je možné provést kontrolu filtrů, zda fungují správně i při přechodů do dalšího období. V případě reportu, který se tvoří v prvním období, kdy jsou data dostupná, je později nutné provést opakované kontroly, zda výchozí hodnoty a filtry byly nastaveny dynamicky a budou zobrazovat správnou hodnotu i při přechodu do dalšího období.

3.3.10.1. Výsledek testovací fáze

Výstupem fáze testování by měl být protokol, ve kterém tester uvede, jaké testy a s jakým výsledkem byly provedeny. V případě zjištěných závad je řešení vráceno zpět vývojáři k odstranění závad, přičemž doba na odstranění závad by měla být dána automaticky na základě nastavení aplikace pro správu firemních projektů anebo smluvních podmínek mezi zadavatelem a externím dodavatelem.

V případě, že testování proběhlo bez závad, tester v protokolu uvede, že řešení je připravené k nasazení.

3.3.10.2. Rizika spojená s testovací fází

Největšími riziky úspěšné testovací fáze je neexistence samostatného a kvalitního testovacího prostředí, absence testera s dostatečnými prostředky, který nebyl zapojen do vývoje reportu a dostatek času na testování.

Pokud přes fázi testování do produkčního prostředí bude nasazen report s vadami, ať už funkčními nebo datovými, hrozí pády nebo chyby v generování reportu, v horším případě i problému modulu, do kterého jsou nový report a jeho součásti integrovány. V případě reportu, který nebude spolehlivě fungovat nebo nebude zobrazovat správná data, může uživatele dojít ke ztrátě důvěry v uvedený report nebo i celému reportingu z tohoto zdroje

3.3.11 Nasazení reportu

3.3.11.1. Nasazení na produkční prostředí

V momentě, kdy je report úspěšně otestován, je možné jej nasadit na produkční prostředí. Při tomto je předveden koncovým uživatelům. Současně s tímto by měla být hotova i uživatelská dokumentace obsahující základní informace např. v jakých intervalech se report generuje, jaká data zobrazuje a co je jejich zdroji. Tyto informace by měly být zaznamenány v seznamu reportů, aby nebyl by opakován vývoj něčeho, co již existuje. Pokud to použitý nástroj reportingu umožňuje, pak je ideální uvést dokumentaci k reportu přímo na stránce, kde se report zobrazuje. Například v Oracle BI Dashboards je to možné to řešit odkazem na dokument obsahující dokumentaci. Uživatel v případě nejasností může okamžitě získat informace na vedlejší záložce a rychle se tak v reportu zorientovat. Tímto se výrazně omezí dotazy na oddělení IT u reportů, které jsou používány zřídka.

Kromě uživatelské dokumentace by měla být hotova kompletní provozní dokumentace, ve které budou popsány zdroje na jednotlivých vrstvách a jejich plnění, použité transformace a cílové umístění reportu.

3.3.11.2. Ustanovení správců jednotlivých reportů

Při nasazení je důležité přidělit k reportu správce, který s reportem sám pracuje. Tento správce průběžně ověřuje správnost dat oproti primárním zdrojům a je tak i kontrolním mechanismem, že uvedený report nepřestal zobrazovat správná data. V případě personální změny by měl existovat automatizovaný kontrolní proces, na základě jehož upozornění by byl k reportu přidělen nový správce. Pokud tento kontrolní proces nebude nastaven, v průběhu let v systému zůstane mnoho

nepoužívaných reportů konzumujících prostor a strojový čas datového skladu žijících si vlastním životem jako kosmický odpad na oběžné dráze planety Země.

3.3.11.3. Pilotní provoz

Pilotní provoz je období po nasazení reportu na produkční prostředí, ve kterém je report standardně generován a využíván, ale jeho datové i funkční spolehlivosti je věnován zvýšený dozor. Délka pilotního provozu záleží zejména na povaze a složitosti reportu, v případě reportu generovaného na denní bázi postačí doba čtrnácti dnů, v případě měsíčního reportu to může být čtvrtletí nebo pololetí, neboť četnost generování takového reportu je nižší a tím pádem je nižší i možnost zachycení chyby v reportu.

3.3.11.4. Vyhodnocení pilotního provozu

Na konci pilotního provozu reportu by mělo následovat vyhodnocení, zda byl report v pilotním provozu spolehlivý a je tedy možné pilotní provoz ukončit nebo během pilotního provozu došlo nebo k problémům a následným opravám a z uvedeného důvodu je nutné pilotní provoz prodloužit.

I v případě úspěšného ukončení pilotního provozu je nutné report zkontrolovat ještě při zvláštních situacích, které nenastaly během pilotního provozu – tím může být například přechod do dalšího měsíce, čtvrtletí, roku nebo fiskálního roku. V takovém případě by kontrolní aktivity měly být naplánovány konkrétním pracovníkům na konkrétní datum a o provedené kontrole by opět měl být učiněn záznam.

3.4. Životní cyklus reportů

Reporty mají svůj životní cyklus stejně jako software. V této kapitole budou uvedeny fáze, které nastávají po předání reportu do běžného provozu, které bylo popsáno v předchozí kapitole.

3.4.1 Pravidelná revize reportů

Revize reportů v pravidelných, například ročních intervalech, zajišťuje stažení reportů, které se již nebudou využívat, protože byly nahrazeny novými reporty, nebo cíl, který sledovaly, ztratil smysl. Dále se ověřuje, jestli se od minulé revize nezměnil zdroj dat a v případě, že ano, jestli bylo tímto způsobem upraveno i generování reportu. Kontrolou by měla projít i práva uživatelů na uvedený report. Do revize reportů by měli být zapojeni správci reportů a tato činnost by jim měla být periodicky plánována a také by jim měl být na to vyhrazen čas, aby revize skutečně proběhla. Samotní správci však nemohou ověřit všechny datové zdroje nebo přiřazení uživatelů, při tomto je nutná součinnost pracovníků oddělení IT.

3.4.1.1. Správa uživatelů reportů

V případě, že jsou ve firmě nastaveny skupiny uživatelů systémově a personální změny probíhají řízeným způsobem, pak by podle pozice nového nebo odcházejícího zaměstnance mělo dojít k automatickému přiřazení nebo odebrání práv na reporty. V rámci zaškolení nového zaměstnance by pak mělo dojít k prezentaci možností využívaných nástrojů reportingu a jemu přidělených reportů, aby se

potenciál využití reportů nesnižoval s odchodem pracovníků, kteří na pozici pracovali v době vývoje a zavedení reportu.

Naopak, pokud přiřazování uživatelů do skupin není systémové, práva na reporty jsou přiřazována konkrétním uživatelům, potom je téměř jisté, že u reportů budou po nějaké době jako příjemci figurovat bývalí pracovníci. Noví zaměstnanci, kteří přijdou na jejich místa, nebudou mít práva na zobrazení reportu nebo nebudou ve skupinách příjemců automatického zasílání reportů nebo se o existenci reportů nedozvědí vůbec.

3.4.1.2. Úprava stávajícího reportu

Společně s vývojem procesů ve firmě se mění i požadavky na reporting. V případě, že se jedná o požadavek na úpravu existujícího reportu, je třeba znovu provést všechny fáze vývoje, jako při vývoji nového přírůstku. Složitost analýzy závisí na složitosti existujícího řešení, pokud se jedná o provozní neagregovaný report, je většinou poměrně jednoduché dodat další atribut a propagovat jej jako nový sloupec v existujícím reportu. Analýza bude náročnější v případě, že cílový report vzniká jako datové tržiště různými transformacemi agregacemi. V případě, že vývojář reportu nevytvořil kvalitní provozní dokumentaci, může být pochopení způsobu přípravy reportu komplikované a časově náročné. V krajních případech může být rozhodnuto, že nežli věnovat mnoho času sledování nejasného zdrojového kódu s nejistým výsledkem, je rychlejší report vyvinout znovu od začátku.

Úprava reportu poskytne zpětnou vazbu, prověří kvalitu kódu a dokumentace reportu, prověří kvalitu vývojáře upravujícího report, zda dosahuje schopností vývojáře, který report vytvořil, v případě interního vývoje to může signalizovat správné předání nebo naopak ztrátu znalostí po předchozím zaměstnanci. Dále se jedná i o prověrku nástroje pro vývoj reportu, jak snadno je formát, ve kterém je report uložen, modifikovatelný, jak snadné je promítnout úpravu datového zdroje do cílového reportu.

Zásadním problémem při úpravě existujícího reportu je granularita dat. Pokud se jedná o nový klíč z dimenze, který by způsobil větší detail dat, není možné takový klíč jednoduše do reportu přidat. Znamenalo by to generování dat od počátku reportu, což v případě historických datových tržišť může být tak velký výkonnostní problém, že by kvůli tomuto muselo být zpožděno nebo odloženo až několik startů plnění datového skladu. Druhou možností je nový klíč přidat, ale pro již existující záznamy vložit jako jeho hodnotu N/A a tohoto řešení si být vědom při prezentaci dat.

3.4.1.3. Vyřazení reportu z provozu

Vyřazením nepotřebných reportů se dávají k dispozici prostředky na generování a ukládání existujících, případně nových reportů. Přesto by vyřazení mělo proběhnout opatrně, nominace na vyřazení by měla vzejít na základě ověření u všech koncových uživatelů, že uvedený report nevyužívají. Kód na tvorbu reportu, případně aplikační struktura, by měly být zálohovány, aby v případě požadavku na obnovení byly k dispozici.

3.5. Syntéza literární rešerše

Souhrn pravidel byl vytvořen na základě syntézy předchozí části práce. Lze je rozdělit na část, kterou může ovlivnit vedení společnosti a na část, kterou mohou ovlivnit vývojáři, analytici a uživatelé reportů.

3.4.2 Souhrn podmínek kladně ovlivňující tvorbu firemního reportingu - odpovědnost vedení společnosti

1. Zavést firemní business slovník, striktně vyžadovat jeho používání při firemní komunikaci a projektové dokumentaci

2. Kvalitu dat primárních systémů a datových skladů nastavit jako jednu z hlavních priorit společnosti, proaktivně opravovat příčiny chyb a činit preventivní opatření

3. Efektivní začleňování uživatelů do skupin, nastavení automatického přiřazování a kontrolních mechanismů v souvislosti s personálními změnami nastavit jako jednu z hlavních priorit společnosti

4. Zavedení shodného vývojového, testovacího a produkčního prostředí

5. Systematicky průběžně modernizovat technologie BI podle potřeby

6. Dbát, aby vývoj prováděli zkušení vývojáři, business analytici a koncoví uživatelé nebo aby probíhal pod jejich vedením

7. Systematicky školit své zaměstnance ve schopnosti kvalifikovaně zadávat požadavky na vývoj, analyzovat tyto požadavky, vést projektová řízení, provádět vývoj

8. Systematicky školit své zaměstnance v uživatelské gramotnosti používání nástrojů reportingu používaných ve společnosti, případně v dolování dat z datového skladu a datových tržišť

9. Systematicky řídit zařazování požadavků na vývoj do jednotlivých vývojových release tak, aby požadavky na reporting byly až v následujícím release po dokončení vývoje v primárních systémech, na kterých jsou závislé¨

10. Vyhodnocovat úspěšnost vývoje jednotlivých požadavků a na základě zjištěných poznatků optimalizovat projektové řízení ve společnosti

11. Nastavit automatický systém kontrol reportů v pilotním a běžném provozu, alokovat čas správců reportu na správu reportů a práv k reportům

12. Budovat důvěru ve vývojový tým a firemní reporting

3.4.3 Souhrn pravidel vývoje firemního reportingu - odpovědnost vývojového týmu

1. Vyvíjet pouze reporty vzešlé ze skutečných potřeb uživatelů

2. Používat firemní business slovník při zadání, specifikaci, komunikaci zúčastněných stran na vývoji, pojmenování sloupců reportu, uživatelské i provozní dokumentaci

3. Minimalizovat zadání na důležitá a okamžitě využitelná fakta

4. Vyvíjet postupně jednotlivé přírůstky podle nejvyššího užitku

5. Dokumentovat každou fázi vývoje do provozní dokumentace

6. Snažit se o jednoduchá a stabilní řešení

7. Při dimenzionálním modelování preferovat typ hvězda, všechny nepotřebné atributy z faktových a dimenzionálních tabulek odstranit z návrhu

8. Řešit kvalitu dat na každém stupni jejich zpracování, poskytovat zpětnou vazbu ohledně
   kvality dat primárních systémů a iniciovat opravu příčin chyb

9. Zachovat a historizovat vstupní data do reportů

10. Dbát při tvorbě databázových struktur, ETL procesů, transformačních a agregačních procedur na optimalizaci

11. Validovat jednotlivé datové vrstvy proti primárním systémům, vyřadit nesmyslné hodnoty, nahradit prázdné hodnoty

12. Tvořit uživatelsky přívětivé reporty

13. Věnovat zvýšenou pozornost fázi testování a minimalizovat případy nasazení řešení s chybami

14. Tvořit kvalitní uživatelskou a provozní dokumentaci, která bude snadno dostupná oprávněným osobám

15. Být vstřícný a ochotný ke koncovým uživatelům v naplnění jejich potřeb, přitom ale plnit požadavky obsažené v zadání a specifikaci

16. Budovat dobré jméno vývojového týmu a důvěru ve firemní reporting

4. Praktická část - vývoj reportu v Oracle BI

Report, jehož vývoj bude v praktické části demonstrován, byl vytvořen po započetí studia odborné literatury k diplomové práci.

4.1. Rozhodnutí o vývoji reportu

Požadavek na vytvoření reportu ohledně úspěšnosti jednotlivých exekutorských společností byl bez dalšího upřesnění ústně sdělen zástupkyní vedoucí právního oddělení. Po obdržení požadavku byla vývojářem inicializována schůzka, kde byly demonstrovány možnosti nástroje Oracle BI, protože zaměstnanci právního oddělení tento ani jiný reportingový nástroj dosud nepoužívali. Poté proběhl krátký brainstorming mezi zaměstnanci právního oddělení, při kterém byl definován základní požadavek na report:

Inkasní společnost na základě klientských smluv v případě neúspěšného mimosoudního inkasního procesu nezaplacené pohledávky postoupí k soudnímu jednání a v případě úspěchu je předají exekutorským společnostem, které zajistí majetek dlužníků a následně úhradu dlužných pohledávek. Inkasní společnost spolupracuje s několika exekutorskými společnostmi, aby co nejvíce naplnila očekávání svých klientů a získala provizi, proto musí sledovat úspěšnost jednotlivých exekutorských společností.

Cílem je vytvořit jednoduchý report, nejlépe ve formě výsečového grafu, který bude členit úspěšnost vymáhání plateb podle jednotlivých exekutorských společností, klienta, kraje působnosti rozhodujícího soudu, druhu vymáhání. Jako filtry jsou požadovány entity Exekutorská společnost, Kraj, Druh vymáhání a Klient. Zobrazení časového hlediska v reportu není požadováno.

Vzhledem k tomu, že při komunikaci pracovníků IT oddělení s pracovníky právního oddělení v minulosti občas vznikala nedorozumění, byly jednotlivé termíny upřesněny podle firemního slovníku. Pojem Platby EXE byl do firemního slovníku doplněn, viz tabulka č. 1.

Pojem	Význam
Kraj	Kraj působnosti rozhodujícího soudu
Exekutor	Název exekutorské společnosti
Klient	Název klienta inkasní společnosti
Druh vymáhání	Rozlišujeme dva druhy vymáhání: Správa: pohledávky našich klientů Odkup: pohledávky, které jsme od klientů odkoupili, a jsou majetkem EOS
Platby EXE	Suma příchozích plateb v CZK od exekutorské společnosti na účet inkasní společnosti

Tabulka č. 1

Na závěr schůzky byla zástupkyně právního oddělení požádána, aby požadavek na vývoj zadala do firemního nástroje na evidenci požadavků na vývoj, aby mohla být vyhodnocena priorita požadavku a případně schválen jeho vývoj.

4.2. Analýza reportu

Prostřednictvím firemní aplikace Alvao byl požadavek na vývoj předán vývojáři. V seznamu reportů bylo ověřeno, že uvedený report již nebyl vyvinut.

V rámci analýzy byla provedena konzultace s pracovnicí finančního oddělení inkasní společnosti. Při tomto bylo zjištěno, že platby do spisů v exekučním řízení přicházejí dvojím způsobem. Buď hromadně z účtů exekutorských společností a poté jsou do jednotlivých spisů přiřazeny na základě variabilního symbolu, který obsahuje zákaznické číslo nebo přichází jednotlivě z účtů jednotlivých dlužníků, přitom do spisu se přiřadí stejným způsobem. Tímto bylo ověřeno, že informace o platbách potřebná k vývoji reportu, je ve standardní tabulce PAYMENTS primární databáze.

V nástroji Enterprise Architect byla provedena analýza schéma primární databáze za účelem zjištění, které tabulky budou potřeba k vývoji požadovaného reportu, viz Příloha 6.1. Při této kontrole bylo zjištěno, že neexistuje vazba mezi předáním spisu Exekutorskému úřadu a platbami, které přišly na účet inkasní společnosti a že tuto vazbu lze odvodit pouze podle data předání spisu Exekutorskému úřadu a datu připsání platby na účet. Na tuto informaci byla upozorněna zadavatelka reportu. Dále byl upozorněn vedoucí IT oddělení, že pro standardní dokončení požadavku je nutné rozšířit struktury a vazby v primární databázi. Vzhledem k tomu, že úpravy procedury na párování plateb z účtů do jednotlivých spisů v interní aplikaci a databázové struktury právního vymáhání byly ve správě rumunského EOS IT Support, předpokládaný vývoj procedury a nových databázových struktur byl možný nejdříve za 3 měsíce.

Po této informaci proběhla schůzka s business analytiky inkasní společnosti, kteří poskytli informaci, že přiřazování plateb k jednotlivým exekutorským společnostem na základě data platby a data předání spisu exekutorské společnosti je velice spolehlivé. Podle jejich názoru je počet chyb v řádu desetin procenta. Toto tvrzení nebylo možné podložit existující statistikou.

Na základě schůzky s business analytiky bylo upřesněno konkrétní zadání reportu. Kromě sledování objemu plateb jednotlivých exekutorských společností bude

v druhé fázi zapotřebí vývoje reportu, který bude informace o platbách spojovat s informací o výši původní exekuční částky v návrhu na exekuci. Úspěšnost exekutorské společnosti bude sledována dle procenta úspěšnosti exekuce, která je dána poměrem objem plateb vymožených v exekuci /částka uvedená v návrhu exekuce. Dále bude užitečné data sledovat i podle měsíce, ve kterém byl konkrétní spis vytvořen.

Vzhledem k souvislosti uvedených informací bylo původní zadání rozšířeno o požadavek, aby v datových strukturách pro report byla dostupná informace o měsíci založení spisu a částce z návrhu exekuce, ačkoliv k jejich zobrazení dojde až v následných reportech. Pojem ‘Částka EXE‘ byl přidán do firemního business slovníku, viz tabulka č. 2.

Částka EXE	Suma částky k exekuci v CZK

Tabulka č.2

Po zvážení všech okolností byla do požadavku přidána poznámka, že zadavatelka je srozuměna s rizikem správnosti přiřazení plateb. Report je pro zcela nezmapované procesy na právním oddělení užitečný, bude mít vypovídající hodnotu, a proto takové riziko akceptuje. Vedoucím IT oddělení inkasní společnosti bylo rozhodnuto o pokračování vývoje řešení, kdy chybějící struktury a vazby budou nahrazeny funkcemi. Současně s tímto bude zadán požadavek na EOS IT Support a po dokončení úpravy procedury na alokaci plateb a potřebných struktur budou funkce nahrazeny tabulkami a vazbami.

Dále bylo prověřeno, zda jsou tabulky z primární databáze ukládány do vrstvy STG0 datového skladu. Kontrolou bylo zjištěno, že tabulka LEGAL_PROPOSAL_INFO v datovém skladu dosud není, bude tedy nejprve třeba ji doplnit do vrstvy STG0.

Název tabulky	Data	Je v datovém skladu?
ACCOUNTS	Případ	ANO
BI_VYRAZENA_CE	Případy vyřazené z reportů	ANO
CLIENTS	Číselník klientů	ANO
COLLECTION_ENTITIES	Obálka případu	ANO
CZ_KRAJ	Číselník soudů	ANO
DYNAMIC_FLAGS	Příznaky definující případ	ANO
LEGAL_CE_PROPOSALS	Žalované případy	ANO
LEGAL_PROPOSAL_INFO	Informace k soudnímu jednání	NE
PAYMENTS	Platby k případům	ANO

Tabulka č. 3

V tabulce BI_VYRAZENA_CE. V tabulce jsou zahraniční pohledávky uskutečněné v České republice. Tyto pohledávky pro svoji povahu není možné vymoci a svými vysokými částkami se dají považovat za extrémní a statisticky zkreslující.

Při ověřování datové kvality v primární databázi bylo zjištěno, že v tabulce LEGAL_CE_PROPOSALS vždy není vyplněna hodnota sloupce LEGAL_INSTITUTION_ID, který je cizím klíčem do číselníku soudů CZ_KRAJ. Dotazem na oddělení právní administrativy bylo zjištěno, že záznamy

vkládají ručně pracovníci administrativy a pole ve formuláři není povinné. Chybějící záznamy budou ošetřeny hodnotou ‘N/A’. Do helpdesku společnosti byl zadán požadavek na prověření, zda je situace s nepovinným polem pro soud v pořádku nebo zda se jedná o chybu.

Report bude dle zadání zobrazovat data v CZK, hodnoty sloupců tabulek:

PAYMENTS.FNC_AMOUNT – částka plateb

LEGAL_PROPOSAL_INFO.OTHERS_AMOUNT – částka exekuce

Hodnoty jsou uvedeny v českých korunách a tak nebude třeba kursové konverze.

Na základě zadání byla sestavena matice dimenzí a faktů, které budou v reportu použity:

Fakta / Dimenze	Částka exekuce v CZK	Částka plateb v CZK
Kraj	X	X
Exekutor	X	X
Klient	X	X
Druh vymáhání	X	X
Měsíc	X	X

Tabulka č. 4

Pracnost vývoje reportu včetně mapování tabulky LEGAL_PROPOSAL_INFO do vrstvy STG0 byla odhadnuta na 16 pracovních hodin, poznámky z fáze analýzy byly formou komentáře zapsány do požadavku.

4.3. Návrh reportu

Nové tabulky potřebné pro report:

```
BI.BI_LEGAL_CE_PROPOSAL_INFO
(
legal_proposal_info_id number(13),
legal_ce_proposal_id number(13),
creted_by varchar2(50),
creation_date date,
last_updated_by varchar2(50),
last_update_date date,
legal_information_type_id number(13),
info_date date,
info_due_date date,
days_no number,
debtor_amount number,
others_amount number,
gat_status varchar2(1),
legal_suits_id number(13),
info_decision_date date
)
tablespace KOLLECTO.BI
```

```
STG0.LEGAL_CE_PROPOSAL_INFO
(
legal_proposal_info_id number(13),
legal_ce_proposal_id number(13),
creted_by varchar2(50),
creation_date date,
last_updated_by varchar2(50),
last_update_date date,
legal_information_type_id number(13),
info_date date,
info_due_date date,
days_no number,
debtor_amount number,
others_amount number,
gat_status varchar2(1),
legal_suits_id number(13),
info_decision_date date
)
tablespace DS.STG0
```

```
STG1.BI_EXEKUTORI_PLATBY
(
 mesic varchar2(7),
 soud_kraj_id number(9),
 exekutor_id number(9),
 client_id number(9),
 druh_vymahani_id number(3),
 castka_exe number,
 platba_exe number,
 row_creation_date date
 ) tablespace DS.STG1
```

```
 STG2.P_INS_F_EXEKUTORI_PLATBY
(
 month_code varchar2(7),
 kraj_id varchar2(9),
 exekutor_id varchar2(9),
 client_code varchar2(9),
 druh_vymahani_id varchar2(3),
 castka_exe number,
 platba_exe number,
 row_creation_date varchar2(10)
 )
tablespace DS.STG2
```

ETL procesy:

KOLLECTO.LEGAL_CE_PROPOSAL_INFO -> BI.BI_LEGAL_CE_PROPOSAL_INFO
zdroj: primární databáze, schéma KOLLECTO -> cíl: primární databáze, schéma BI
přenos: procedura v BI.BI_LOAD
perioda plnění: denní
agregace: 1:1

BI.BI_LEGAL_CE_PROPOSAL_INFO -> STG0.LEGAL_CE_PROPOSAL_INFO
zdroj: primární databáze, schéma BI -> cíl: datový sklad, schéma STG0
přenos: Kettle, EOS STG0 LEGAL – LOAD
perioda plnění: denní
agregace: 1:1

Tabulky vrstvy STG0 -> STG1.BI_EXEKUTORI_PLATBY

zdroj: datový sklad, schéma STG0 -> cíl: datový sklad, schéma STG1
přenos: procedura STG1.STG1_LOAD.P_BI_EXEKUTORI_PLATBY
perioda plnění: denní
agregace: MESIC - to_char(trunc(COLLECTION_ENTITIES,CREATION_DATE),'yyyy-mm')
KRAJ – CZ_KRAJ.SOUD_KRAJ_ID
EXEKUTOR – COLLECTION_ENTITIES.STATUS_ID
KLIENT – CLIENTS.CLIENT_ID
DRUH_VYMAHANI – CLIENTS.DP
CASTKA_EXE – LEGAL_PROPOSAL_INFO.OTHERS_AMOUNT
PLATBY_EXE - PAYMENTS.FNC_AMOUNT -funkce

Použité funkce:

Název: GET_CE_PAYMENTS_BETWEEN
Vstupní parametry: P_CEID – COLLECTION_ENTITIES.CEID
P1_DATE – datum od
P2_DATE – datum do
Návratová hodnota: NUMBER – suma plateb připsaných do spisu v době od - do pro daný CEID

Název: GET_EXE_NAVRH_PODANI
Vstupní parametry: P_CEID – COLLECTION_ENTITIES.CEID
P_FLAG – number : 1 - identifikátor návrhu exekuce
2 - identifikátor exekučního řízení
Návratová hodnota: NUMBER – podle P_FLAG identifikátor návrhu exekuce nebo identifikátor příslušného exekučního řízení.

STG1.BI_EXEKUTORI_PLATBY -> STG2.F_EXEKUTORI_PLATBY
zdroj: datový sklad, schéma STG1 -> cíl: datový sklad, schéma STG2
přenos: STG2_LOAD. P_INS_F_EXEKUTORI_PLATBY
perioda plnění: denní
agregace: 1:1

Úprava repository Oracle BI:

Nová faktová tabulka: F_EXEKUTORI_PLATBY
Nové dimenzionální tabulky: D_KRAJ_F_EXEKUTORI_PLATBY
D_DRUH_VYMAHANI_F_EXEKUTORI_PLATBY
D_EXEKUTOR_F_EXEKUTORI_PLATBY
D_TIME_MONTH_F_EXEKUTORI_PLATBY
D_CLIENTS_F_EXEKUTORI_PLATBY
Nové skupiny: Právní
Nový uživatel: Právník

Úprava dashboards Oracle BI:

Panel: Právní

Stránka: Exekutoři platby

Právo: skupina Právní

Zobrazení: výsečový graf

Hierarchie: Exekutor, Druh vymáhání, Klient

Výzva panelů: Kraj, Exekutor, Druh vymáhání, Klient

4.4. Implementace reportu

V této kapitole budou popsány kroky, které byly podle návrhu provedeny na vývojovém prostředí.

4.4.1. Vytvoření a plnění datových struktur

V primární databázi ve schématu KOLLECTO bylo uživateli BI přidáno právo na select z tabulky KOLLECTO.LEGAL_PROPOSAL_INFO. Ve schématu BI byla vytvořena tabulka BI.BI_LEGAL_PROPOSAL_INFO shodné struktury, jako tabulka zdrojová tabulka KOLLECTO.LEGAL_PROPOSAL_INFO.

Plnění tabulky BI.BI_LEGAL_PROPOSAL_INFO bylo přidáno do procedury BI.BI_LOAD.LOAD_BI_SCHEMA v balíčku BI_LOAD. Z důvodu otestování byla tabulka BI.BI_LEGAL_PROPOSAL_INFO ručně naplněna kódem přidaným do procedury BI.BI_LOAD.LOAD_BI_SCHEMA. V datovém skladu ve vrstvě STG0 byla vytvořena tabulka STG0.LEGAL_PROPOSAL_INFO. Kódy SQL k uvedené části jsou v Příloze 6.2.

V nástroji Kettle Spoon byla upravena transformace EOS STG0 LEGAL – LOAD. Bylo vytvořeno mapování zdrojové tabulky z primární databáze BI.BI_LEGAL_PROPOSAL_INFO na cílovou tabulku datového skladu STG0. LEGAL_PROPOSAL_INFO, viz Příloha 6.3. Po připojení byla ručně spuštěna transformace EOS STG0 LEGAL – LOAD, došlo k naplnění tabulek právní oblasti vrstvy STG0 včetně STG0. LEGAL_PROPOSAL_INFO.

Byla vytvořena funkce GET_CE_PAYMENTS_BETWEEN s parametry P_CEID, P1_DATE, P2_DATE, návratovou hodnotou datového typu NUMBER, která na základě identifikátoru konkrétního spisu a dat ohraničujících

období předání konkrétní inkasní společnosti vrátí sumu plateb v CZK, které byly v uvedeném období připsány do spisu.

Náklady vnitřního dotazu funkce byly ověřeny v okně Exekuční plán nástroje PL/SQL Developer. Náklady 8 byly vyhodnoceny jako přiměřené a bez nutnosti další optimalizace, přestože se nejedná o kompletní náklad spojený s voláním funkce.

Funkce byla otestována spuštěním a kontrolou správnosti výstupní hodnoty s parametry (879586, 1.6.2014, 30.6.2014; 880423, 10.6.2014, 9.7.2014; 880435, 30.6.2014, 30.7.2014). Po úspěšném testu byla přidána do balíčku s názvem LEGAL uloženém ve schématu STG1 datového skladu. Kód funkce STG1.LEGAL.GET_CE_PAYMENTS_BETWEEN a exekuční plán vnitřního dotazu funkce jsou v Příloze 6.4.

Byla vytvořena funkce GET_EXE_NAVRH_PODANI s parametry P_CEID, P_FLAG a návratovou hodnotou datového typu NUMBER, která dle identifikátoru konkrétního spisu vrátí v případě hodnoty parametru P_FLAG = 1 vrátí identifikátor návrhu exekuce, v případě hodnoty parametru P_FLAG = 2 vrátí identifikátor příslušného exekučního řízení.

Ověření nákladů funkce bylo provedeno v okně Exekuční plán nástroje PL/SQL Developer pro všechny tři dotazy, které jsou ve funkci použity. Maximální náklady dotazů byly dohromady 10, byly vyhodnoceny jako přiměřené a bez nutnosti další optimalizace, přestože se nejedná o kompletní náklad spojený s voláním funkce.

Funkce byla otestována spuštěním a kontrolou správnosti výstupní hodnoty s parametry (879586, 1; 879586, 2; 880423, 1; 880423, 2; 880435, 1; 880435, 2).

Po úspěšném testu byla přidána do balíčku s názvem LEGAL uloženém ve v databázovém schématu STG1 datového skladu. Kód funkce STG1.LEGAL.GET_EXE_NAVRH_PODANI je v příloze v Příloze 6.5, exekuční plán vnitřního dotazu funkce v Příloze 6.6.

Byla vytvořena tabulka BI_EXEKUTORI_PLATBY ve schématu STG1, kód vytvoření tabulky je v Příloze 6.7.

Byla vytvořena procedura P_BI_EXEKUTORI_PLATBY s parametrem P_CREATION_DATE pro plnění agregované tabulky STG1.BI_EXEKUTORI_PLATBY.

Náklady procedury byly ověřeny v okně Exekuční plán nástroje PL/SQL Developer, celkové náklady činily 20 446, což nejsou z hlediska ostatních databázových dotazů nepřiměřené náklady, byť skutečné náklady procedury STG1.STG1_LOAD.P_BI_EXEKUTORI_PLATBY budou vyšší vzhledem k použití funkce GET_CE_PAYMENTS_BETWEEN a funkce GET_EXE_NAVRH_PODANI. Byl zde však nalezen prostor pro optimalizaci, neboť data z tabulky COLLECTION_ENTITIES byla získána metodou FULL TABLE SCAN, tedy načtením všech dat tabulky, ačkoliv případů v exekuční fázi vymáhání je pouze zlomek.

Nad tabulkou STG0.COLLECTION_ENTITIES byl vytvořen index COLLECTION_ENTITIES_IDX, byly přepočítány statistiky tabulky a znovu byly ověřeny náklady procedury STG1.STG1_LOAD.P_BI_EXEKUTORI_PLATBY. V exekučním plánu pro přístup k datům tabulky STG0.COLLECTION_ENTITIES byla nyní použita přístupová metoda INDEX RANGE SCAN, tedy čtení dat tabulky na základě použití indexu COLLECTION_ENTITIES_IDX, což náklady dotazu procedury snížilo na 489.

Tabulka STG0.COLLECTION_ENTITIES byla vymazána, dále byla ručně spuštěna transformace EOS STG0 – LOAD. Čas plnění tabulky STG0.COLLECTION_ENTITIES oproti času předchozí plnění, kdy nad tabulkou ještě nebyl vytvořen index COLLECTION_ENTITIES_IDX, byl o 15,7 s rychlejší. Prodloužení času plnění STG0.COLLECTION_ENTITIES bylo vyhodnoceno jako nevýznamné, vzhledem k tomu, že vytvořený index COLLECTION_ENTITIES_IDX nebude využit jen při generování tvořeného reportu, ale zejména první čtyři indexované sloupce budou využity i v dotazech pro jiné reporty či ad-hoc dotazy.

Procedura byla spuštěna ručně, plnění cílové tabulky STG1. BI_EXEKUTORI_PLATBY proběhlo bez chyb v čase 5:13 min.

Procedura byla přidána do balíčku s názvem STG1_LOAD uloženém ve schématu STG1 datového skladu. Volání procedury bylo přidáno do řídící procedury balíčku STG1.STG1_LOAD.MAIN.

SQL kód funkce STG1.STG1_LOAD.P_BI_EXEKUTORI_PLATBY je v Přílohách 6.8 a 6.9. Exekuční plány vnitřního dotazu před a po vytvoření indexu COLLECTION_ENTITIES_IDX včetně kódu SQL tvorby indexu a přepočítání statistik tabulky STG0.COLLECTION_ENTITIES je v Příloze 6.10 .

Byla vytvořena tabulka F_EXEKUTORI_PLATBY ve schématu STG2, kód vytvoření tabulky je v Příloze 6.11.

Byla vytvořena procedura P_INS_F_EXEKUTORI_PLATBY s parametrem P_INDATE, která smaže a vyplní záznamy cílové tabulky STG2.F_EXEKUTORI_PLATBY. Procedura byla spouštěna ručně, proběhla bez chyb v čase 9,8 s.

Procedura byla přidána do balíčku s názvem STG2_LOAD uloženém ve schématu STG2 datového skladu. Volání procedury bylo přidáno do řídící procedury balíčku STG2.STG2_LOAD.MAIN. SQL kód vytvoření procedury je v Příloze 6.12.

4.4.2. Vytvoření prezentační vrstvy v Oracle BI

Na fyzickou vrstvu Oracle BI byla v nástroji AnylticsWeb přes volbu File -> Import do logické složky KOLLECTO_FACT importována tabulka F_EXEKUTORI_PLATBY, viz Příloha 6.13.

Na fyzické vrstvě byla příkazem Create -> Alias z tabulky D_KRAJ vytvořena dimenzionální tabulka D_KRAJ_F_ EXEKUTORI_PLATBY.

Na fyzické vrstvě byla příkazem Create -> Alias z tabulky D_CLIENTS vytvořena dimenzionální tabulka D_CLIENTS_F_ EXEKUTORI_PLATBY.

Na fyzické vrstvě byla příkazem Create -> Alias z tabulky D_EXEKUTOR vytvořena dimenzionální tabulka D_EXEKUTOR_F_ EXEKUTORI_PLATBY.

Z tabulky D_TIME_MONTH byla příkazem Create -> Alias vytvořena dimenzionální tabulka D_TIME_MONTH_F_ EXEKUTORI_PLATBY.

Z tabulky D_DRUH_VYMAHANI byla příkazem Create -> Alias vytvořena dimenzionální tabulka D_DRUH_VYMAHANI_F_ EXEKUTORI_PLATBY.

Na fyzické vrstvě byly společně označeny tabulky:

F_EXEKUTORI_PLATBY
D_KRAJ_F_EXEKUTORI_PLATBY
D_DRUH_VYMAHANI_F_EXEKUTORI_PLATBY
D_EXEKUTOR_F_EXEKUTORI_PLATBY
D_TIME_MONTH_F_EXEKUTORI_PLATBY
D_CLIENTS_F_EXEKUTORI_PLATBY

a přes volbu View -> Selected tables Only byly zobrazeny na plátně fyzické vrstvy. Dimenzionální tabulky byly spojeny s faktovou tabulkou přes sloupce:

D_EXEKUTOR_F_EXEKUTORI_PLATBY.EXEKUTOR_ID-> F_EXEKUTORI_PLATBY. EXEKUTOR_ID
D_KRAJ_F_EXEKUTORI_PLATBY.KRAJ_ID -> F_EXEKUTORI_PLATBY. KRAJ_ID

D_DRUH_VYMAHANI_F_EXEKUTORI_PLATBY.DRUH_VYMAHANI_ID ->
F_EXEKUTORI_PLATBY. DRUH_VYMAHANI_ID

D_TIME_MONTH_F_EXEKUTORI_PLATBY.MONTH_CODE->
F_EXEKUTORI_PLATBY.MONTH_CODE

D_CLIENTS_F_EXEKUTORI_PLATBY. CLIENT_CODE -> F_EXEKUTORI_PLATBY. CLIENT_CODE

Spojení dimenzionálních a faktové tabulky je v Příloze 6.14. Repository bylo uloženo jako konzistentní.

Všechny označené tabulky byly přetaženy do prostředního bazénu nástroje AnylticsWeb - business vrstvy Oracle BI. U sloupců CASTKA_EXE a PLATBY_EXE tabulky F_EXEKUTORI_PLATBY byly hodnoty agregovány, jako výchozí agregace byla nastavena funkce Sum. Repository bylo uloženo jako konzistentní.

Všechny označené tabulky byly přetaženy do levého bazénu nástroje AnylticsWeb - prezentační vrstvy Oracle BI. Repository bylo uloženo jako konzistentní.

Byla vytvořena dimenze Dim Exekutori Platby. Jako hlavní úroveň byl nastaven Total s vlastností Grand Total.

K úrovni Total byla vytvořena podřízená úroveň Exekutor ze sloupce EXEKUTOR tabulky D_EXEKUTOR_F_EXEKUTORI_PLATBY.

K úrovni Exekutor byla vytvořena podřízená úroveň DP ze sloupce DRUH_VYMYHANI tabulky D_DRUH_VYMAHANI_F_EXEKUTORI_PLATBY

K úrovni DP byla vytvořena podřízená úroveň Klient ze sloupce NAME tabulky D_CLIENTS_F_EXEKUTORI_PLATBY

Repository bylo uloženo jako konzistentní. Dimenze Dim Exekutori Platby je zobrazena v Příloze 6.15.

4.4.3. Vytvoření reportu v Oracle BI

V Oracle BI Answers byly na plátno reportu přidány a přejmenovány sloupce:

F_EXEKUTORI_PLATBY.MONTH_CODE, přejmenováno na Měsíc
F_EXEKUTORI_PLATBY.CASTKA_EXE, přejmenováno na Částka EXE
F_EXEKUTORI_PLATBY.PLATBY_EXE, přejmenováno na Platby EXE
D_KRAJ_F_EXEKUTORI_PLATBY.KRAJ, přejmenováno na Kraj
D_EXEKUTOR_F_EXEKUTORI_PLATBY.EXEKUTOR, přejmenováno na Exekutor
D_DRUH_VYMAHANI_F_EXEKUTORI_PLATBY.DRUH_VYMAHANI, přejmenováno na Druh vymáhání
D_CLIENTS_F_EXEKUTORI_PLATBY.CLIENT_NAME, přejmenováno na Klient

.

U sloupců reportu Druh vymáhání, Kraj, Exekutor a Klient byla nastavena volba Obsahuje výzvu, aby tyto sloupce reagovaly na filtry z výzvy panelů, viz Příloha 6.16. Jako zobrazení reportu byl zvolen výsečový graf, přičemž do pole hodnot byl přidán sloupec Platby EXE. Report byl uložen do sdílené složky Legal pod názvem VV_EXEKUTORI_PLATBY.

Byla vytvořena nová výzva panelů, jako filtry výzvy byly použity sloupce:

D_KRAJ_F_EXEKUTORI_PLATBY.KRAJ, přejmenováno na Kraj
D_EXEKUTOR_F_EXEKUTORI_PLATBY.EXEKUTOR, přejmenováno na Exekutor
D_CLIENTS_F_EXEKUTORI_PLATBY.CLIENT_NAME, přejmenováno na Klient
D_DRUH_VYMAHANI_F_EXEKUTORI_PLATBY.DRUH_VYMAHANI, přejmenováno na Druh vymáhání

Výzva panelů byla uložena pod názvem VP_VV_EXEKUTORI_PLATBY do sdílené složky Legal.

V Oracle BI Dashboards byl vytvořen nový panel s názvem Právní. Na panelu Právní byla vytvořena stránka Exekutoři platby. Na stránku byla přidána Sekce 1, do Sekce 1 byla vložena výzva panelů VP_VV_EXEKUTORI_PLATBY, report VV_EXEKUTORI_PLATBY a objekt Odkaz nebo obrázek 1. Vložení prvků do Sekce 1 je zobrazeno v Příloze 6.17.

V prvku Odkaz nebo obrázek 1 byla uložena cesta na stránku firemní wikipedie s popisem reportu, viz Příloha 6.18. Stránka Exekutoři platby byla uložena.

4.4.4. Přiřazení uživatelů reportu

V Oracle BI Administration tool – Analytics Web byl vytvořen uživatel Právník. Byla vytvořena skupina Právní, do této skupiny byl přiřazen uživatel Právník.

Proběhlo přihlášení uživatelem Právník do Oracle BI Dashboards z důvodu, aby se zobrazil mezi seznamem uživatelů a skupin, ze kterého se vybírá při přiřazování práv k panelům, stránkám a reportům.

Již pod uživatelem Administrator ve správě Oracle BI Dashboards bylo skupině Právní nastaveno právo číst na panel Právní, viz Příloha 6.19.

Proběhlo přihlášení uživatelem Právník do Oracle BI Dashboards a panel Právní byl nastaven jako výchozí panel pro uživatele Právník. Uživateli Právník se současně s tím zobrazily ostatní panely, kde existuje přiřazení práva Everyone.

4.4.5. Dokumentace reportu

Uživatelská dokumentace k reportu byla uložena na stránce podnikové wikipedie a zpřístupněna na stránce reportu formou odkazu.

Jako programátorská dokumentace slouží několik zdrojů, které se týkají datových struktur i podmínek generování reportu. V helpesku u zadaného úkolu byl jako příloha vložen dokument typu Word obsahující finální informace ze zadání požadavku, analýzy a návrhu. V nástroji Enterprise Architect byla do modelu vrstvy STG0 přidána tabulka STG0. LEGAL_PROPOSAL_INFO. Do modelu vrstvy STG1 byla přidána tabulka STG1.BI_EXEKUTORI_PLATBY, k tabulce byla vložena poznámka, ve které byly uvedeny hlavní informace ohledně způsobu generování dat tabulky. Do modelu vrstvy STG2 byla přidána tabulka STG2.F_EXEKUTORI_PLATBY.

Do seznamu reportů byl přidán záznam o nově vyvinutém reportu:

Název reportu	Účel reportu	Panel BI	Stránka BI	Správce reportu
Exekutoři platby	Přehled plateb exekutorských společností	Právní	Exekutoři platby	Přemysl Jindra

Tabulka č. 5

4.5. Testování reportu

Při implementaci byly vyzkoušeny všechny funkční prvky reportu, tedy:

- drill-down hierarchií Exekutor, DP, Klient – viz Přílohy 6.20, 6.21 a 6.22
- použití filtrů výzvy panelů Kraj, Exekutor, Klient a Druh vymáhání
- kombinace filtrů a dril drill-down
- funkčnost odkazu na popis reportu na podnikové wikipedii
- export reportu do Excelu a PDF

Pro otestování datové části byly vytvořen kontrolní skript, kterým se kontrolují shodné sumy Platby EXE a Částka EXE mezi jednotlivými tabulkami od primární databáze do vrstvy STG2 datového skladu:

KOLLECTO.LEGAL_PROPOSAL_INFO -> BI. LEGAL_PROPOSAL_INFO

BI.LEGAL_PROPOSAL_INFO -> STG0. LEGAL_PROPOSAL_INFO

STG0.LEGAL_PROPOSAL_INFO -> STG1.BI_EXEKUTORI_PLATBY

STG1.BI_EXEKUTORI_PLATBY -> STG2.F_EXEKUTORI_PLATBY

Jednotlivé SQL dotazy skriptu i celý skript byl vyzkoušen vývojářem, výsledky SQL dotazů zobrazovaly souhlasné hodnoty a celý skript byl spuštěn bez chyb. Skript viz Přílohy 6.23 a 6.24.

Pro uživatelské i datové testování byly report a skript předány business analytikovi se zkušenostmi z užívání reportů v Oracle BI a dotazování SQL k testování prostřednictvím aplikace Alvao.

4.6. Nasazení reportu

Po ukončení testů s vyjádřením testujícího, že report splňuje požadavky zadání, byl požadavek předán zpět vývojáři k nasazení do produkčního prostředí.

Na základě uložených SQL kódů při tvorbě datových struktur byly postupně vytvořeny tabulky:

BI. LEGAL_PROPOSAL_INFO ve schématu BI primární databáze
STG0. LEGAL_PROPOSAL_INFO ve schématu STG0 datového skladu
STG1.BI_EXEKUTORI_PLATBY ve schématu STG1 datového skladu
STG2.F_EXEKUTORI_PLATBY ve schématu STG2 datového skladu

V nástroji Kettle Spoon byla upravena transformace EOS STG0 LEGAL – LOAD. Bylo vytvořeno mapování zdrojové tabulky BI.BI_LEGAL_PROPOSAL_IN FO z primární databáze na cílovou tabulku STG0. LEGAL_PROPOSAL_INFO z dat ového skladu.

Stav balíčku BI.BI_LOAD v primární databázi byl zkontrolován oproti stavu balíčku na vývojovém prostředí v nástroji PL/SQL Developer volbou Compare object, při porovnání bylo věřeno, že balíčky se liší pouze částí kódu související s plněním tabulky BI. LEGAL_PROPOSAL_INFO. Balíček BI.BI_LOAD primární databáze

byl zálohován, balíček BI.BI_LOAD z vývojové databáze byl přenesen do primární databáze a zkompilován.

Stav balíčku STG1.STG1_LOAD v produkčním datovém skladu byl zkontrolován oproti stavu balíčku ve vývojovém datovém skladu v nástroji PL/SQL Developer volbou Compare object. Při porovnání bylo věřeno, že tyto dva balíčky se liší pouze přítomností procedury P_BI_EXEKUTORI_PLATBY a jejího volání na vývojovém prostředí. Balíček STG1.STG1_LOAD produkčního datového skladu byl zálohován, balíček STG1.STG1_LOAD z vývojové databáze byl přenesen do produkčního datového skladu a zkompilován.

Stav balíčku STG2.STG2_LOAD v produkčním datovém skladu byl zkontrolován oproti stavu balíčku ve vývojovém datovém skladu v nástroji PL/SQL Developer volbou Compare object. Při porovnání bylo věřeno, že tyto dva balíčky se liší pouze přítomností procedury P_INS_F_EXEKUTORI_PLATBY a jejího volání na vývojovém prostředí. Balíček STG2.STG2_LOAD produkčního datového skladu byl zálohován, balíček STG2.STG2_LOAD z vývojové databáze byl přenesen do produkčního datového skladu a zkompilován.

Skripty a procedury pro plnění tabulek byly spuštěny ručně, datové struktury byly naplněny.

Produkční repository Oracle BI Kollecto.rpd a vývojové repository Kollecto_DEV.rpd byly dány do offline režimu. Produkční repository Oracle BI Kollecto.rpd bylo zálohováno.

Z fyzické vrstvy Kollecto_DEV.rpd byly kopírovány a vloženy na fyzickou vrstvu Kollecto.rpd tabulky:

F_EXEKUTORI_PLATBY
D_KRAJ_F_EXEKUTORI_PLATBY
D_EXEKUTOR_F_EXEKUTORI_PLATBY
D_DRUH_VYMAHANI_F_EXEKUTORI_PLATBY
D_CLIENTS_F_EXEKUTORI_PLATBY

Mezi tabulkami byly vytvořeny vazby. Repository Kollecto.rpd bylo uloženo jako konzistentní. Označené tabulky na fyzické vrstvě repository byly označeny a

přetaženy do business vrstvy. U sloupců CASTKA_EXE a PLATBY_EXE tabulky F_EXEKUTORI_PLATBY byly hodnoty agregovány, jako výchozí agregace byla nastavena funkce Sum. Hierarchie Dim Exekutori Platby byla kopírována z Kollecto_DEV.rpd a uložena do business vrstvy repository Kollecto.rpd. Repository Kollecto.rpd bylo uloženo jako konzistentní. Označené tabulky byly přetaženy do prezentační vrstvy produkčního prostředí. Repository Kollecto.rpd bylo uloženo jako konzistentní.

V Oracle BI Administration tool – Analytics Web v repository Kollecto.rpd byl vytvořen uživatel Právník. Byla vytvořena skupina Právní, do této skupiny byl přiřazen uživatel Právník.

Na produkčním prostředí byla provedena záloha Web Catalog. Na vývojovém prostředí byla kopírována složka Právní, tato byla vložena do Web Catalog na produkčním prostředí.

Repository Kollecto.rpd a Kollecto_DEV.rpd byly dány do online stavu.

Po importu reportu na produkční prostředí bylo prověřeno jeho zobrazení, funkce drill-down, filtrů výzvy panelů. Po úspěšném ověření funkčnosti byl report předán do pilotního provozu prostřednictvím aplikace Alvao.

5. Závěr

Cílem této práce bylo vytvoření souhrnu pravidel firemního reportingu na základě syntézy znalostí získaných studiem odborné literatury, školením specialistů a vlastních zkušeností při tvorbě firemního reportingu. Vedlejším cílem mělo být uvedení jednotlivých fází vývoje reportu a rizik, která tyto fáze nejvíce ohrožují.

Sestavený souhrn pravidel tvorby firemního reportingu byl rozdělen na zásady, které by měly být dodržovány vedením společnosti a zásady, které by měli dodržovat vývojáři reportingu.

Pravidla pro vedení společnosti vytvářejí vhodné prostředí, aby vývoj mohl probíhat standardně, bez negativních dopadů do produkčních systémů a vyvinutá řešení byla co nejvíce užitečná, spolehlivá a bezpečná. Vedení společnosti též vytváří základní motivaci pro proaktivní spolupráci při tvorbě a využívání firemního reportingu a důvěru v reportovaná data.

Dodržování pravidel pro vývojáře by mělo zajistit, že vyvinutá řešení budou co nejvíce odpovídat požadavkům zadavatelů a termíny dodání řešení se budou blížit odhadu pracnosti řešení po fázi analýzy. Vyvinuté reporty by měly svou pravdivostí, jednoduchostí a spolehlivostí podporovat důvěru ve firemní reporting a vývojový tým.

V práci byly popsány jednotlivé fáze vývoje reportu. V každé fázi byla uvedena hlavní rizika ohrožující danou fázi a řešení, kterými se dají tato rizika předcházet nebo minimalizovat.

V praktické části byl proveden vývoj reportu v prostředí Oracle BI s důrazem na dodržování pravidel shrnutých v teoretické části. Pokud se praktická část bude hodnotit kriticky oproti pravidlům pro vývoj reportu, potom je zde několik bodů, které je třeba zmínit jako prohřešky proti pravidlům nebo prostor pro zlepšení:

1. Vývojář neměl k dispozici tři, ale pouze dvě srovnatelná prostředí, z čehož jedno sloužilo jako vývojové a testovací prostředí zároveň a druhé jako produkční.

2. Na všech fázích vývoje kromě hlavní části fáze testování se podílel pouze jeden vývojář, a to i ve fázi zadání a analýzy, což bylo dáno velikostí firmy a absencí specialistů projektového řízení.

3. Při analýze byl požadavek rozšířen o další sloupce, které v původním zadání nebyly požadovány a v reportu ani nebyly zobrazeny. Vzhledem k tomu, že to nezpůsobilo navýšení pracnosti reportu a v průběhu vývoje reportu byl business analytiky založen požadavek na nový report za využití těchto sloupců, bylo zahrnutí požadovaných sloupců do související datové struktury správné a efektivní.

4. Při vývoji reportu bylo k plnění struktur využito funkcí, což v prostředí SQL může mít negativní dopad na výkonnost dotazů. Toto bylo dáno neexistující vazbou mezi exekutorskou společností a platbou připsanou na účet inkasní společnosti v době výkonu exekuce v primárním systému. Vývojář na možné negativní důsledky v oblasti výkonnosti upozornil vedoucího pracovníka a zadavatele reportu, bylo však rozhodnuto o pokračování ve vývoji reportu a dodatečném vývoji vazby v primárním systému. Použité funkce byly optimalizovány za použití indexu.

5. Zdrojové tabulky nebyly historizovány, vzhledem k častým změnám v primárním systému a důvěře v databázové zdroje a neustálenému procesu v primárním systému

Při celkovém pohledu na praktickou část lze konstatovat, že pravidla vývoje z teoretické části práce byla dodržena. Důraz byl kladen na kvalitu analýzy a návrhu a

optimalizaci ETL procesů. Následná uživatelská i programátorská dokumentace se dají považovat na dostatečně kvalitní na revizi či rozšíření stávajícího stavu. Ačkoliv vývoj reportu prováděl vývojář se zkušenostmi z oblasti BI, při analýze a vývoji nastaly situace rozhodování, jak uvedenou problematiku řešit systémově správně.

Z důvodů uvedených v předchozím odstavci lze definovat hlavní myšlenku závěru diplomové práce. Investice části pracovního času analytiků a vývojářů do osvojení si zásad tvorby firemního reportingu se následně vrátí ve vyšší efektivitě při vývoji, kvalitě řešení a nižších nákladech vývoje.

Vzdělávání lze docílit formou externího školení nebo interních workshopů, když zkušenější pracovníci školí ty méně zkušené. Vzhledem k personálním změnám v podniku a životnímu cyklu firemních aplikací by to měl být nikdy nekončící proces.

6. Seznam literatury a použitých zdrojů

1. FIBÍROVÁ, Jana. *Reporting: moderní metoda hodnocení výkonnosti uvnitř firmy*. 2. aktualiz. vyd. Praha: Grada, 2003. Účetnictví a daně (Grada). ISBN 80-247-0482-X.
2. TYRYCHTR, Jan. *Business intelligence*. Vyd. 1. V Praze: Česká zemědělská univerzita, Provozně ekonomická fakulta, 2014. ISBN 978-80-213-2516-6.
3. URBAN, Jan. *Řízení lidí v organizaci: personální rozměr managementu*. 2., rozš. vyd. Praha: Wolters Kluwer Česká republika, 2013. Vzdělávání dospělých. ISBN 978-80-7357-925-8.
4. LABERGE, Robert. *Datové sklady: agilní metody a business intelligence*. 1. vyd. Brno: Computer Press, 2012. ISBN 978-80-251-3729-1.
5. POUR, Jan. Self-service Business Intelligence . Systémová integrace, 2014.
6. HUGHES, Ralph. Agile data warehousing project management: business intelligence systems using Scrum and XP. 1st ed. Waltham, MA: Morgan Kaufmann, 2013. ISBN 9780123964632.
7. POUR, Jan, Miloš MARYŠKA a Ota NOVOTNÝ. *Business intelligence v podnikové praxi*. 1. vyd. Praha: Professional Publishing, 2012. ISBN 978-80-7431-065-2.
8. NOVOTNÝ, Ota, Jan POUR a David SLÁNSKÝ. Business Intelligence: *Jak využít bohatství ve vašich datech. Praha: Grada Publishing a.s.*, 2005, 256 s. ISBN 978- 80-247-6685-0.
9. ZYKA Ondřej. *Datová kvalita- základ úspěšného BI.* Business Intelligence fórum 2012, Praha http://www.cssi.cz/cssi/datova-kvalita-zaklad-uspesneho-bi
10. TONA, Olgerta; CARLSSON, Sven. An Evaluation of Potential Benefits of Mobile BI. In: *European Conference on Information Management and Evaluation*. Academic Conferences International Limited, 2013. p. 185.
11. VIKTOR MAYER-SCHÖNBERGER AND KENNETH CUKIER. *Big data: a revolution that will transform how we live, work and think*. London: Murray, 2013. ISBN 9781848547933.
12. INMON William H. *Building the data warehouse*. 4th ed. Indianapolis, Ind.: Wiley, c2005, xxviii, 543 p. ISBN 07-645-9944-5.

13. ECKERSON, Wayne. Performance dashboards: measuring, monitoring, and managing your business. Hoboken: Wiley, c2006, xviii, 301 s. ISBN 978-0-471-72417-9.

14. KIMBALL, Ralph a Margy ROSS. *Data warehouse toolkit: the definitive guide to dimensional modeling*. Third edition. Indianapolis: John Wiley & Sons, 2013. ISBN 978-1-118-53080-1.

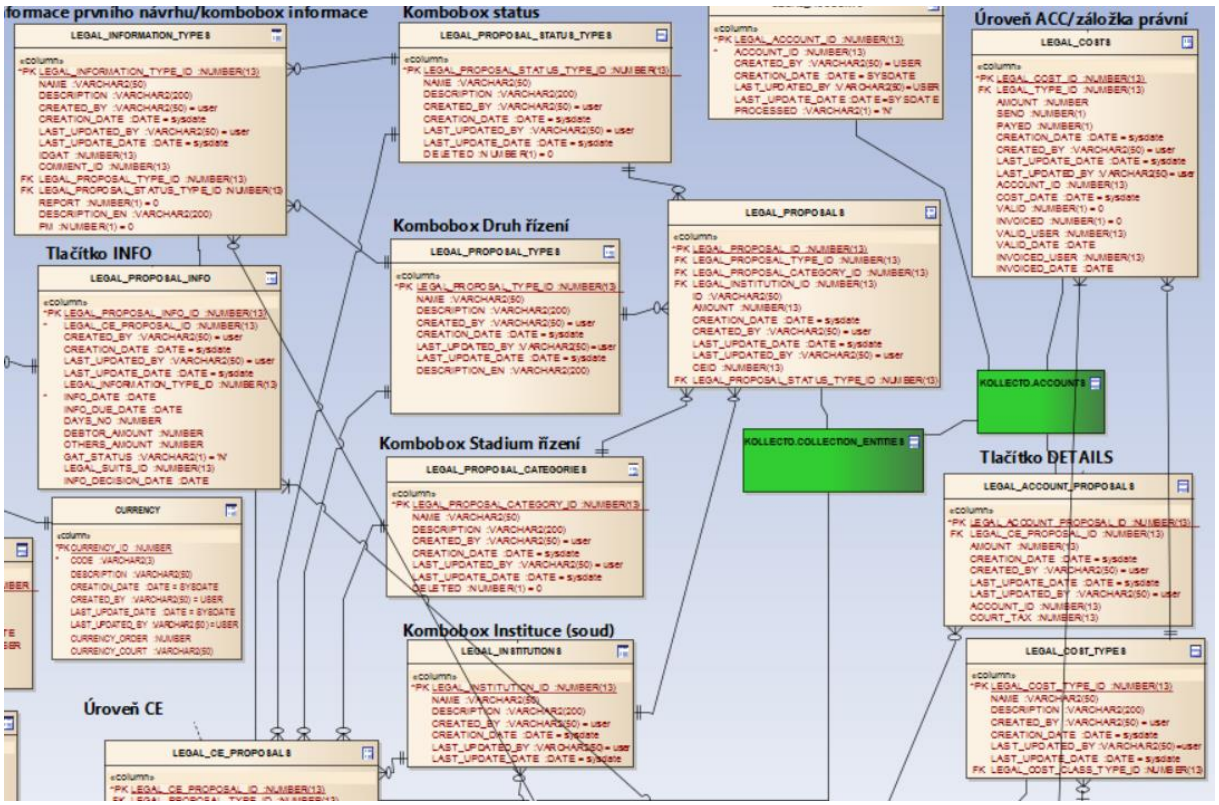
15. DYCHÉ, Jill a Evan LEVY. *Customer data integration: reaching a single version of the truth*. Hoboken, N.J.: John Wiley & Sons, c2006. ISBN 9780471916970.

16. YEOH William, KORONIOS Andy, *Critical Success Factors for Business Intelligence Systems*. Journal of Computer Information Systems Vol. 50, Iss. 3, 2010

17. GEMIGNANI, Zach, Chris GEMIGNANI, Richard GALENTINO a Patrick Jude SCHUERMANN. *Efektivní analýza a využití dat*. 1. vydání. Překlad Jiří Huf. Brno: Computer Press, 2015. ISBN 978-80-251-4571-5.

7. Přílohy

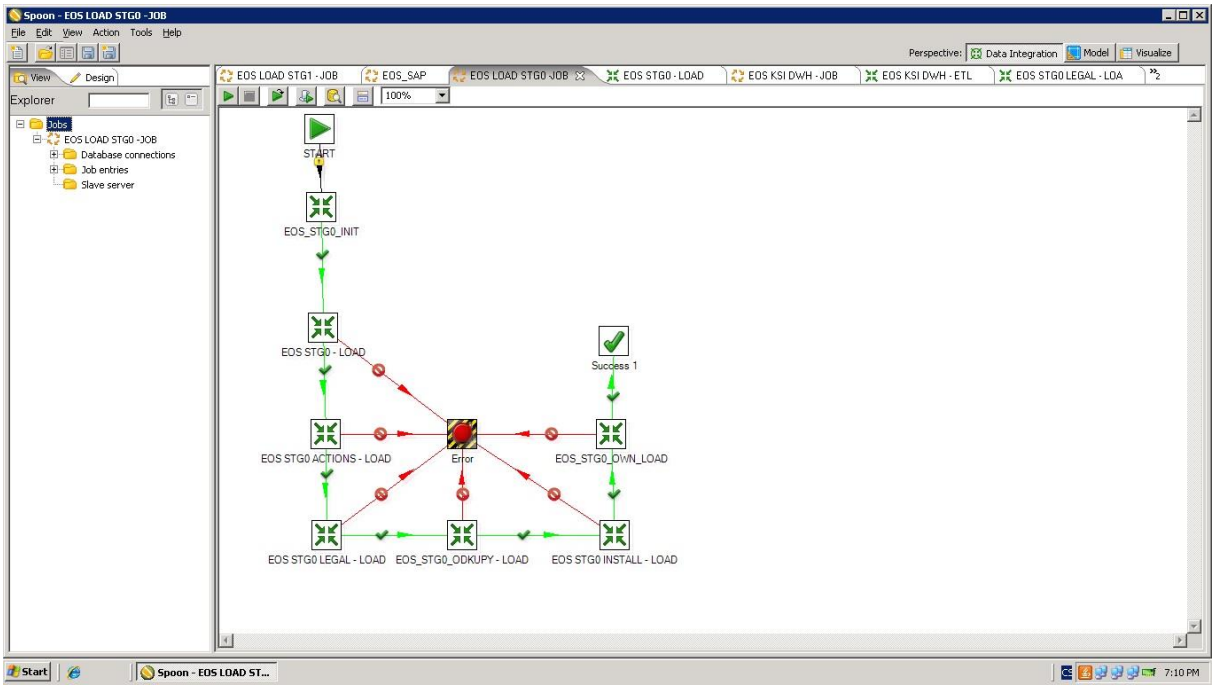
Příloha 6.1 - Schéma primární databáze KOLLECTO - právní oblast

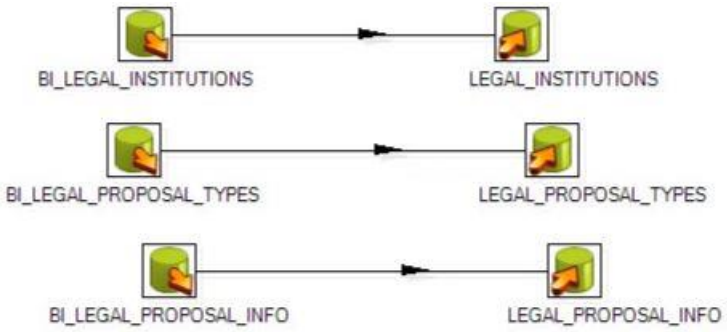


Příloha 6.2 - Příprava databázových struktur pro přenos dat mezi primární databází a vrstvou STG0 datového skladu

```
        Grant select on legal_proposal_info to BI


        create table bi_legal_proposal_info tablespace BI as
        select li.*,  trunc(sysdate) as  creation_date from kollecto.legal_proposal_info
li;



        insert into bi.bi_legal_proposal_info(
                                            legal_proposal_info_id,
                                            legal ce proposal_id,
                                            creted by,
                                            creation_date,
                                            last_updated_by,
                                            last update date,
                                            legal information_type_id,
                                            info date,
                                            info due_date,
                                            days_no,
                                            debtor_amount,
                                            others amount,
                                            gat status,
                                            legal suits id,
                                            info_decision_date
                                            )
        select legal_proposal_info_id, legal_ce_proposal_id,
               creted by, creation date, last updated by,
               last update date, legal information type id,
               info date, info due date, days no, debtor amount,
               others_amount, gat_status, legal_suits_id,
               info_decision_date, d_creation_date
        from kollecto.legal_proposal_info;
        commit;


        create table legal_proposal_info tablespace stg0
        as select * from bi.bi_legal_proposal_info@KOLLECTO;
```

Příloha 6.3 - Mapování zdrojové tabulky v primární databázi a cílové tabulky v datovém skladu v ETL nástroji Kettle





Příloha 6.4 - Kód funkce GET_CE_PAYMENTS_BETWEEN a její náklady v exekučním plánu PL/SQL Developer

```
function get ce payments between(
                                  p_ceid in collection_entities.ceid%type,
                                  p1_date in payments.payment_date%type,
                                  p2_date in payments.payment_date%type
                                 ) return number
is
n_amount number;
begin

  select sum(p.fnc_amount) into n_amount
  from accounts a, payments p
  where a.account id = p.account_id
  and a.ceid = p_ceid
  and p.payment_date between p1_date and p2_date;

return nvl(n_amount,0);
exception


  when others then return 0;
 end;
```

Description	Object owner	Object name	Cost
SELECT STATEMENT, GOAL = ALL_ROWS			8
SORT AGGREGATE			
FILTER			
NESTED LOOPS			8
INDEX RANGE SCAN	KOLLECTO	ACCOUNTS_CEID_ACC_I	3
TABLE ACCESS BY INDEX ROWID BATCHED	KOLLECTO	PAYMENTS	5
INDEX RANGE SCAN	KOLLECTO	PAYMENTS_CD_ACCOUNT_ID_TYPEII	2

Příloha 6.4 - Kód funkce GET_CE_PAYMENTS_BETWEEN a její náklady v exekučním plánu PL/SQL Developer

```
function get ce payments between(
                                  p_ceid in collection_entities.ceid%type,
                                  p1_date in payments.payment_date%type,
                                  p2_date in payments.payment_date%type
                                 ) return number
is
n_amount number;
begin

  select sum(p.fnc_amount) into n_amount
  from accounts a, payments p
  where a.account id = p.account_id
  and a.ceid = p_ceid
  and p.payment_date between p1_date and p2_date;

return nvl(n_amount,0);
exception


  when others then return 0;
 end;
```

Description	Object owner	Object name	Cost
SELECT STATEMENT, GOAL = ALL_ROWS			8
SORT AGGREGATE			
FILTER			
NESTED LOOPS			8
INDEX RANGE SCAN	KOLLECTO	ACCOUNTS_CEID_ACC_I	3
TABLE ACCESS BY INDEX ROWID BATCHED	KOLLECTO	PAYMENTS	5
INDEX RANGE SCAN	KOLLECTO	PAYMENTS_CD_ACCOUNT_ID_TYPEII	2

Příloha 6.5 - Kód funkce GET_EXE_NAVRH_PODANI

```
create or replace function get_exe_navrh_podani
(p_ceid in legal_ce_proposals.ceid%type, p_flag in number) return number

is
n_legal_ce_proposal_id  legal_ce_proposals.legal_ce_proposal_id%type;
begin
  if p_flag = 1
    then
      select max(lce.legal ce proposal_id) into n_legal_ce_proposal_id
      from legal ce proposals lce
      where lce.ceid = p_ceid
      and exists (select 1
                    from  legal_proposal_info lpi
                    where lpi.legal information type id = 138
                    and lpi.legal_ce_proposal_id = lce.legal_ce_proposal_id
                  );
  elsif
    p_flag = 2
    then
       select max(lce.legal ce proposal_id) into n_legal_ce_proposal_id
      from legal ce proposals lce
      where lce.ceid = p_ceid
      and lce.legal_proposal_type_id = 43
      and exists (select 1
                    from  legal proposal info lpi
                    where lpi.legal information type id = 139
                    and lpi.legal_ce_proposal_id = lce.legal_ce_proposal_id
                  );

      if n_legal_ce_proposal_id is null
        then
          select max(lce.legal ce proposal_id) into n_legal_ce_proposal_id
          from legal_ce_proposals lce
          where lce.ceid = p_ceid
          and lce.legal_proposal_type_id = 21;
      end if;
  end if;
return n legal_ce_proposal_id;
exception
  when others then return null;
end;
```

Příloha 6.6 - Náklady funkce GET_EXE_NAVRH_PODANI v exekučním plánu PL/SQL Developer

Description	Object owner	Object name	Cost
SELECT STATEMENT, GOAL = ALL_ROWS			7
SORT AGGREGATE			
NESTED LOOPS SEMI			7
INDEX RANGE SCAN	STG0	LEGAL_CE_PROPOSALS_CE_FK	3
INDEX RANGE SCAN	STG0	LEGAL_PROPOSAL_INFO_TYPE_FK	2

Description	Object owner	Object name	Cost
SELECT STATEMENT, GOAL = ALL_ROWS			5
SORT AGGREGATE			
NESTED LOOPS SEMI			5
INDEX RANGE SCAN	STG0	LEGAL_CE_PROPOSALS_CE_SLOZ	3
INDEX RANGE SCAN	STG0	LEGAL_PROPOSAL_INFO_TYPE_FK	2

Description	Object owner	Object name	Cost
SELECT STATEMENT, GOAL = ALL_ROWS			3
SORT AGGREGATE			
FIRST ROW			3
INDEX RANGE SCAN (MIN/MAX)	STG0	LEGAL_CE_PROPOSALS_CE_SLOZ	3

Příloha 6.7 - Vytvoření tabulky BI_EXEKUTORI_PLATBY

```
create table BI_EXEKUTORI_PLATBY
(
 mesic varchar2(7),
 soud_kraj_id number(9),
 exekutor_id number(9),
 client_id number(9),
 druh vymahani id number(3),
 castka exe number,
 platby_exe number,
 row_creation_date date
 )
 tablespace STG1
```

Příloha 6.8 - Kód procedury P_BI_EXEKUTORI_PLATBY část. 1

```
        procedure P_BI_EXEKUTORI_PLATBY (p_creation_date in date)
        is
        v_inDate varchar2(10) := to_char(p_creation_date,'dd.mm.yyyy');
        begin
        etl_log.p_start_log('P_BI_EXEKUTORI_PLATBY',    '    BI_EXEKUTORI_PLATBY    ',
v_inDate,'STG1');
        execute immediate 'truncate table bi_exekutori_platby';

        insert into   bi_exekutori_platby (mesic,soud_kraj_id,exekutor_id, client_id,
druh_vymahani_id,castka_exe,platby_exe,row_creation_date)

        select mesic,
               nvl(leg.soud_kraj_id,'N/A') as soud_kraj_id,
               leg.status_id as ekekutor_id,
               leg.client_id,
               leg.druh vymahani id
               sum(leg.castka exe) as castka exe,
               sum(get ce payments_between(leg.ceid,
               leg.navrh_exe,
               add_months(leg.navrh_exe,1))) as platby_exe,
               d_creation_date as row_creation_date,

        from (
               select cel.mesic,
                      cel.druh_vymahani_id,
                      cel.client_id,
                      cel.ceid,
                      cel.status id,
                      nvl((select sum(lpi.others amount)
                            from legal proposal info lpi
                            where lpi.legal_information_type_id = 138
                            and lpi.legal_ce_proposal_id =
                                     cel.legal_ce_proposal_id1),0) as castka_exe,
                      nvl((select max(lpi.info date)
                             from legal proposal info lpi
                             where lpi.legal_information_type_id = 138
                          and               lpi.legal_ce_proposal_id               =
                       cel.legal_ce_proposal_id1),d_creation_date) as navrh_exe,
                      (select ck.soud kraj_id
                       from cz kraj ck
                       where ck.legal institution id =
                                (select  lce.legal_institution_id
                                   from legal_ce_proposals lce
                                             where      lce.legal_ce_proposal_id      =
                                     cel.legal_ce_proposal_id2)) as soud_kraj_id
               from (
                     select to char(trunc(ce.creation_date),'yyyy-mm') as mesic
                          cl.dp as druh_vymahani_id,
                          ce.client_id,
                          ce.ceid,
                          ce.status id,
                          get_exe_navrh_podani(ce.ceid,1) as legal_ce_proposal_id1,
```

Příloha 6.9 - Kód procedury P_BI_EXEKUTORI_PLATBY část. 2

```
                    select to_char(trunc(ce.creation_date),'yyyy-mm') as mesic
                           cl.dp as druh_vymahani_id,
                           ce.client_id,
                           ce.ceid,
                           ce.status_id,
                           get_exe_navrh_podani(ce.ceid,1)as legal_ce_proposal_id1,
                    get_exe_navrh_podani(ce.ceid,2) as legal_ce_proposal_id2
                   from collection entities ce, clients cl
                   where nvl(ce.status, 'Y') like 'EXNE%'
                   and cl.client_id = ce.client_id
                   and trunc(ce.creation_date) > to_date('31.12.2007','dd.mm.yyyy')
                   and not exists(
                                  select 1
                                  from dynamic flags df
                                  where df.ceid = ce.ceid
                                  and df.flag_id = 1602
                                  )
                    and not exists (
                                   select 1
                                   from bi vyrazena ce cez
                                   where ce.ceid = cez.ceid
                                   )
               ) cel
          ) leg
     group by leg.mesic, (leg.soud_kraj_id,'N/A'), leg.status_id, leg.client_id,
leg.druh_vymahani_id;

       commit;

       etl log.p_end_log('  P_BI_EXEKUTORI_PLATBY  ',  '  BI_EXEKUTORI_PLATBY  ',  d_
inDate,'OK' );


       exception
       when others then
       rollback;

       alert := 'Pro den: '||INDATE||' vznikla chyba v procedure P_BI_EXEKUTORI_PLATBY
pri plnení tabulky BI_EXEKUTORI_PLATBY: '||SQLERRM;

       etl_log.p_end_log(' P_BI_EXEKUTORI_PLATBY ', ' BI_EXEKUTORI_PLATBY ', 0, ALERT
);

       end P_BI_EXEKUTORI_PLATBY;
```

Příloha 6.10 - Exekuční plány procedury P_BI_EXEKUTORI_PLATBY a vytvoření indexu COLLECTION_ENTITIES_IDX při optimalizaci procedury P_BI_EXEKUTORI_PLATBY

Description	Object owner	Object name	Cost
SELECT STATEMENT, GOAL = ALL_ROWS			20446
FILTER			
TABLE ACCESS FULL	KOLLECTO	CZ_KRAJ	3
TABLE ACCESS BY INDEX ROWID	KOLLECTO	LEGAL_CE_PROPOSALS	3
INDEX UNIQUE SCAN	KOLLECTO	LEGAL_CE_PROPOSALS_PK	2
SORT AGGREGATE			
TABLE ACCESS BY INDEX ROWID BATCHED	KOLLECTO	LEGAL_PROPOSAL_INFO	4
INDEX RANGE SCAN	KOLLECTO	LEGAL_PROPOSAL_INFO_TYPE_FK	3
SORT AGGREGATE			
TABLE ACCESS BY INDEX ROWID BATCHED	KOLLECTO	LEGAL_PROPOSAL_INFO	4
INDEX RANGE SCAN	KOLLECTO	LEGAL_PROPOSAL_INFO_TYPE_FK	3
INDEX RANGE SCAN	KOLLECTO	CLIENTS_CLIENT_ID	2
HASH GROUP BY			20446
VIEW	KOLLECTO		20141
NESTED LOOPS ANTI			14520
HASH JOIN RIGHT ANTI			14520
INDEX RANGE SCAN	KOLLECTO	DYNAMIC_FLAGS_CEID_IDX	32
TABLE ACCESS FULL	KOLLECTO	COLLECTION_ENTITIES	14487
INDEX UNIQUE SCAN	KOLLECTO	BI_VYRAZENA_CE	0

```
create index COLLECTION_ENTITIES_IDX
on COLLECTION_ENTITIES (
                              ceid, client_id, trunc(creation_date),
                              nvl(status, 'Y'),
                              to_char(trunc(creation_date),'yyyy-mm')
                        )
tablespace STG0_IDX;
analyze table COLLECTION_ENTITIES compute statistics;
```

Description	Object owner	Object name	Cost
SELECT STATEMENT, GOAL = ALL_ROWS			489
FILTER			
TABLE ACCESS FULL	STG0	CZ_KRAJ	3
TABLE ACCESS BY INDEX ROWID	STG0	LEGAL_CE_PROPOSALS	3
INDEX UNIQUE SCAN	STG0	LEGAL_CE_PROPOSALS_PK	2
SORT AGGREGATE			
TABLE ACCESS BY INDEX ROWID BATCHED	STG0	LEGAL_PROPOSAL_INFO	4
INDEX RANGE SCAN	STG0	LEGAL_PROPOSAL_INFO_TYPE_FK	3
SORT AGGREGATE			
TABLE ACCESS BY INDEX ROWID BATCHED	STG0	LEGAL_PROPOSAL_INFO	4
INDEX RANGE SCAN	STG0	LEGAL_PROPOSAL_INFO_TYPE_FK	3
INDEX RANGE SCAN	STG0	CLIENTS_CLIENT_ID	2
HASH GROUP BY			489
VIEW	STG0		184
NESTED LOOPS ANTI			120
HASH JOIN RIGHT ANTI			120
INDEX RANGE SCAN	STG0	DYNAMIC_FLAGS_CEID_IDX	32
INDEX RANGE SCAN	STG0	COLLECTION_ENTITIES_IDX	76
INDEX UNIQUE SCAN	STG0	BI_VYRAZENA_CE	0

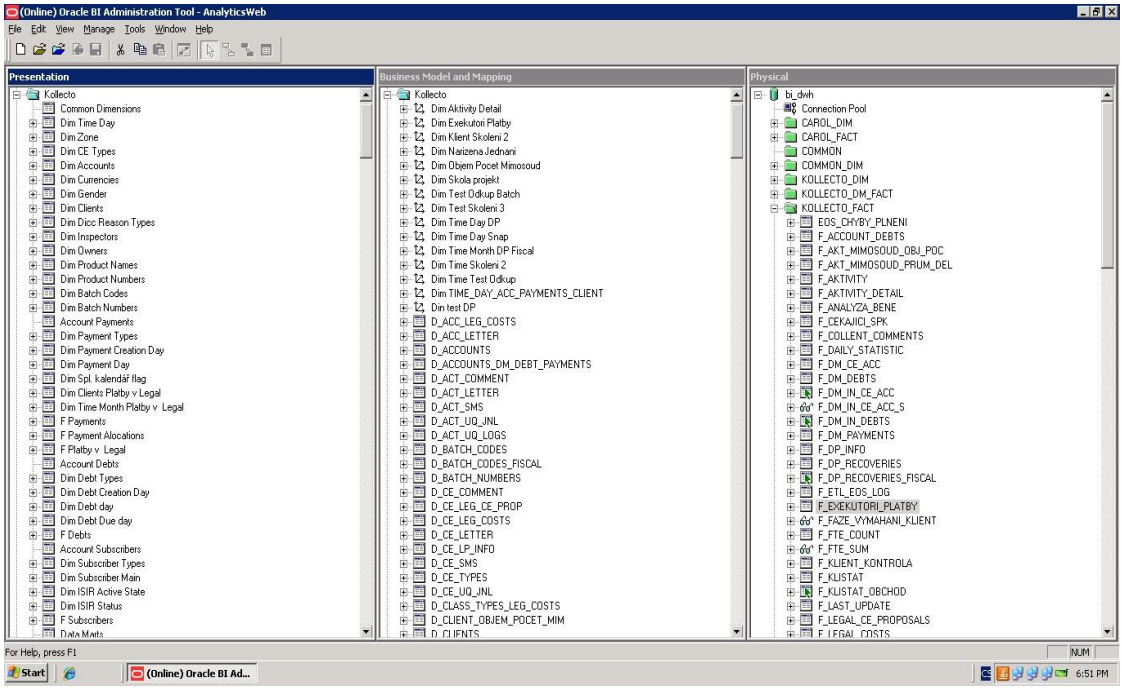
Příloha 6.11 - Vytvoření tabulky P_INS_F_EXEKUTORI_PLATBY

```
create table P_INS_F_EXEKUTORI_PLATBY
(
 month_code varchar2(7),
 kraj id varchar2(9),
 exekutor id varchar2(9),
 client_code varchar2(9),
 druh_vymahani_id varchar2(3),
 castka_exe number,
 platby exe number,
 row_creation_date varchar2(10)
 )
 tablespace STG2
```

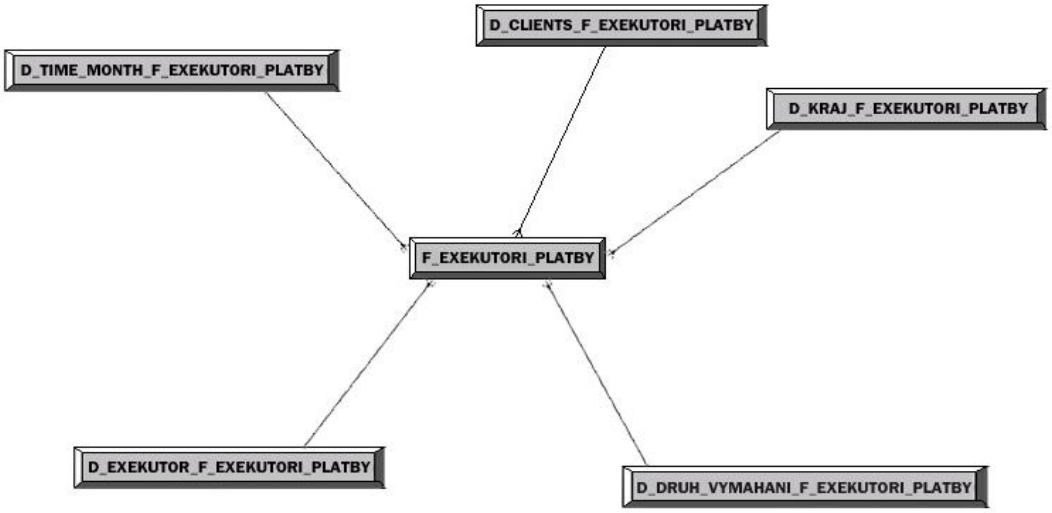
Příloha 6.12 - Kód procedury P_INS_F_EXEKUTORI_PLATBY

```
procedure P_INS_F_EXEKUTORI_PLATBY(p_indate in varchar2)
  is
  v alert varchar2(2000);
  begin

  etl_log.p_start_log('P_INS_F_EXEKUTORI_PLATBY','F_EXEKUTORI_PLATBY',v_indate,'F');
  execute immediate 'truncate table F_EXEKUTORI_PLATBY';

        insert into F EXEKUTORI PLATBY(
        month code, kraj id, exekutor id, client code, druh_vymahani_id,
        castka_exe, platby_exe, row_creation_date
        )
        select  se.mesic as month_code,
                to char(se.soud kraj id) as kraj id,
                to char(se.exekutor id) as exekutor id,
                to char(se.client id) as client code,
                to_char(se.druh_vymahani_id) as druh_vymahani_id,
                sum(se.castka_exe) as castka_exe,
                sum(se.platby_exe) as platby_exe,
                p indate
        from  bi exekutori platby se
        group  by     se.mesic,  to_char(se.soud_kraj_id),  to_char(se.exekutor_id),
to_char(se.client_id), to_char(se.druh_vymahani_id);

        commit;
        etl_log.p_end_log ('P_INS_F_EXEKUTORI_PLATBY', 'F_EXEKUTORI_PLATBY', 1, 'OK' );

        exception
        when others then
        rollback;
        v alert := 'Pro den: '||p indate||' vznikla chyba v procedure
                    P INS F EXEKUTORI PLATBY pri plnení tabulky
                    F_EXEKUTORI_PLATBY: '||SQLERRM;

        etl_log.p_end_log ('P_INS_F_EXEKUTORI_PLATBY', 'F_EXEKUTORI_PLATBY', 0, v_alert
);
        end P_INS_F_EXEKUTORI_PLATBY;
```

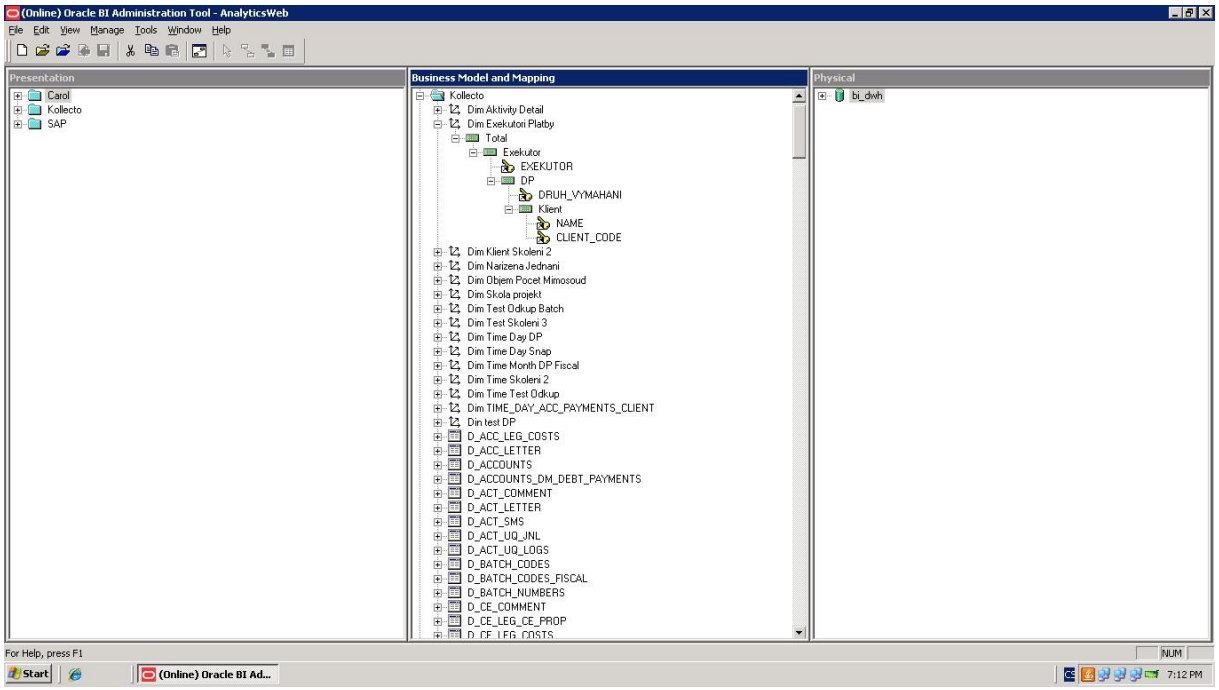
Příloha 6.13 - Import faktové tabulky F_EXEKUTORI_PLATBY na fyzickou vrstvu Oracle BI



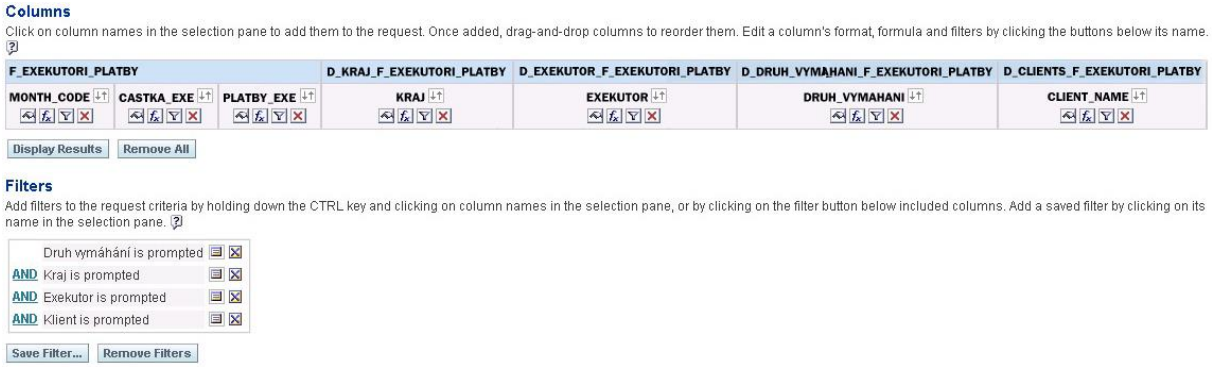
Příloha 6.14 - Spojení dimenzionálních tabulek a faktové tabulky na fyzické vrstvě Oracle BI



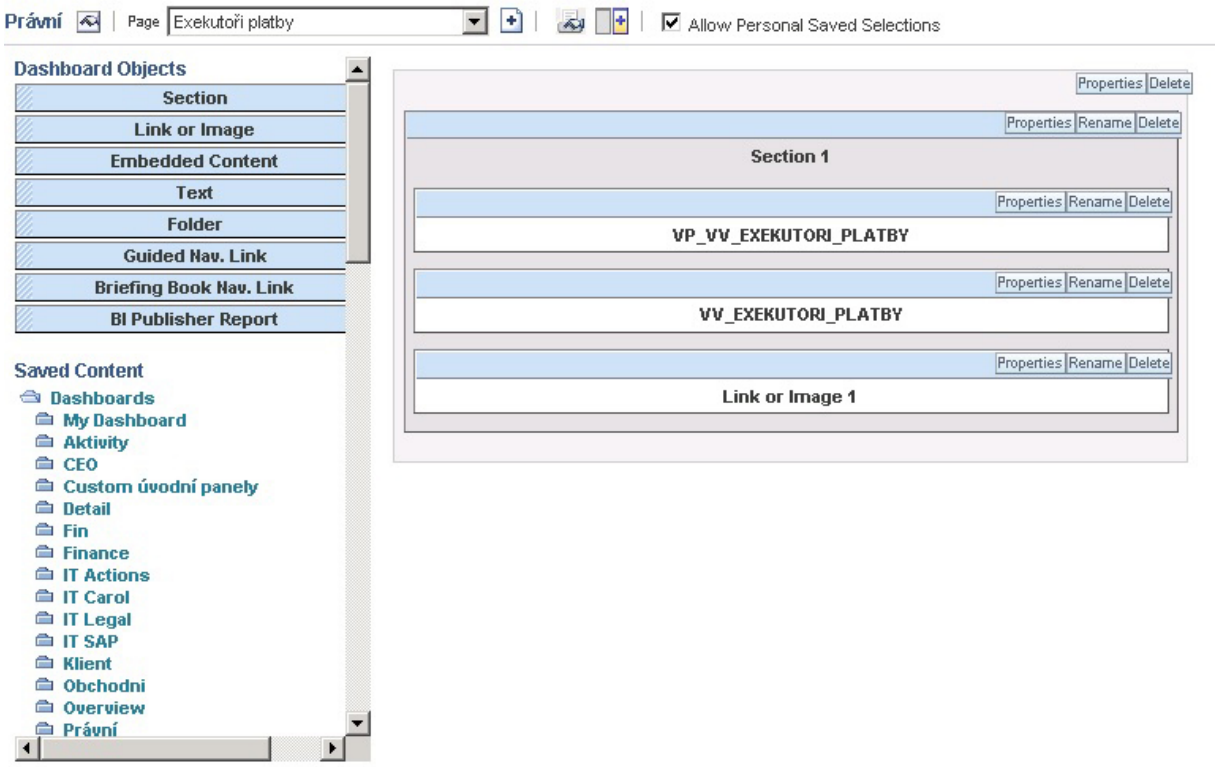
Příloha 6.15 - Vytvoření hierarchické dimenze na business vrstvě Oracle BI



Příloha 6.16 - Tvorba reportu v Answers Oracle BI



Příloha 6.17 - Složení částí reportu v Dashboards Oracle BI



Příloha 6.18 – Popis reportu na firemní wikipedii

report: **Exekutoři platby**

cíl reportu: report zobrazuje platby vymožené exekutorskými společnostmi, které spolupracují s inkasní společností, rozdělené podle typu vymáhání, exekutorské společnosti, kraje trvalého bydliště dlužníka a klienta za celou dobu činnosti inkasní společnosti

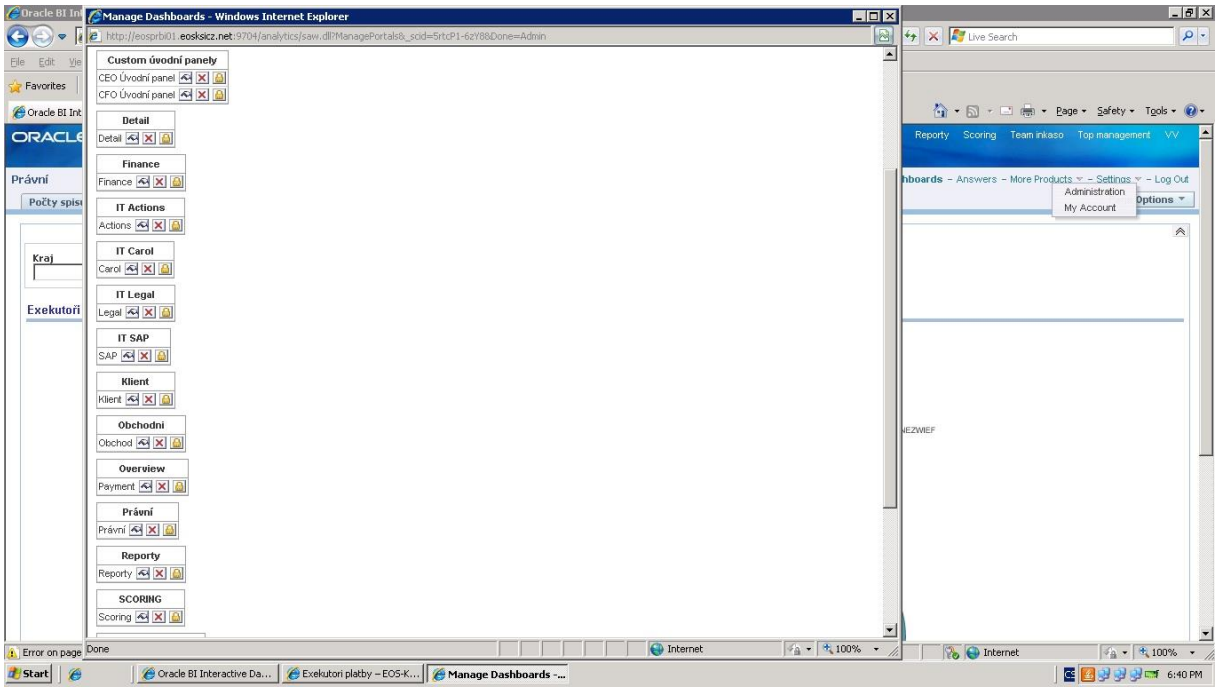
perioda reportu: měsíční

sloupce reportu:

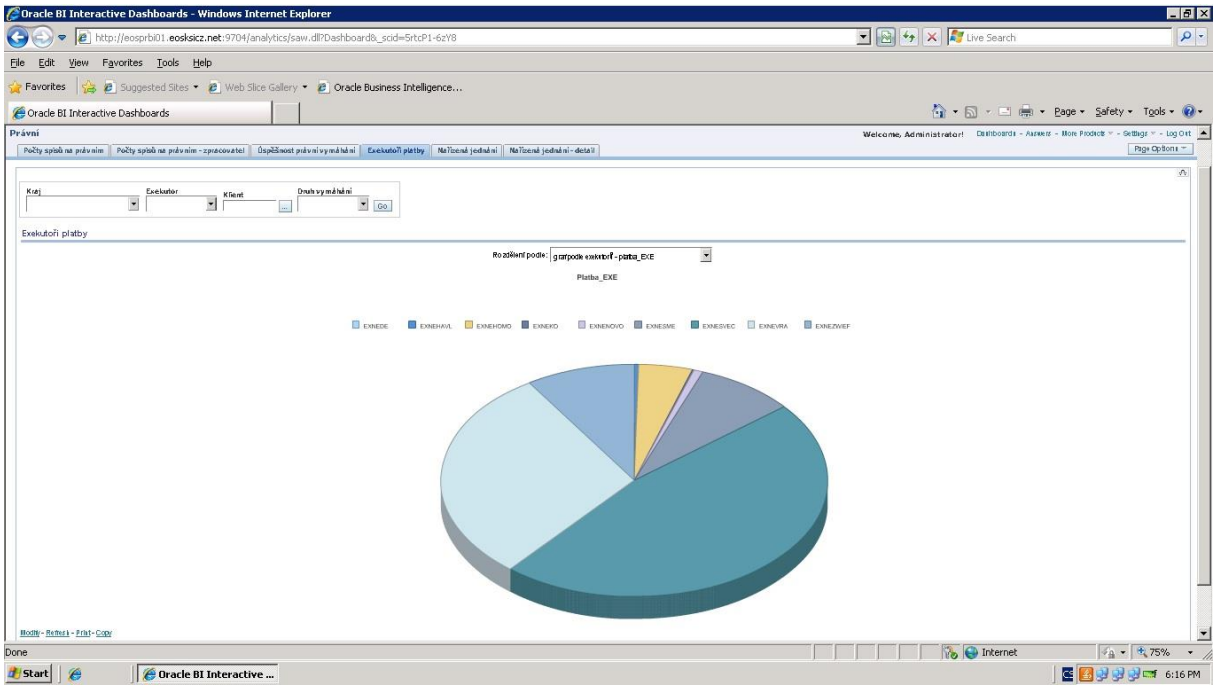
SLOUPEC	VÝZNAM SLOUPCE	ZDROJ
Klient	název klienta	Clients.Name
Exekutor	Název exekutorské společnosti	Collection_entities.Status
Druh vymáhání	Druh vymáhání – odkup, správa	Clients.Dp
Částka EXE	Celková částka exekuce	Legal_proposal_info.Lpi_others_amount
Platby EXE	Vymožené platby exekutorskou společností	Payments.Fnc_amount
Kraj	Kraj trvalého bydliště dlužníka	Cz_kraj.Soud_kraj

zadavatel reportu: Michaela Novotná, právní oddělení
příjemce reportu: právní oddělení
správce reportu: Přemysl Jindra, business analytik
tvůrce reportu: Ondřej Moravec, oddělení IT
report nasazen do provozu: 10.2.2015
poslední modifikace:

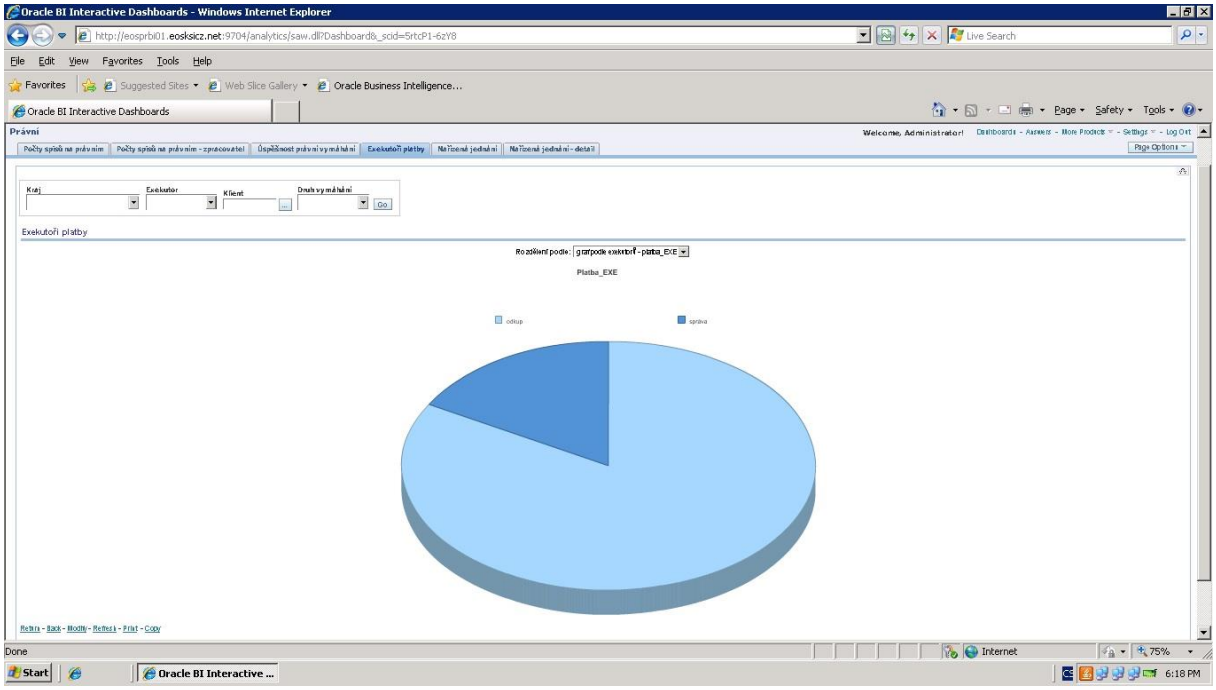
Příloha 6.19 – Přidělení práv k reportu v Oracle BI Dashboards



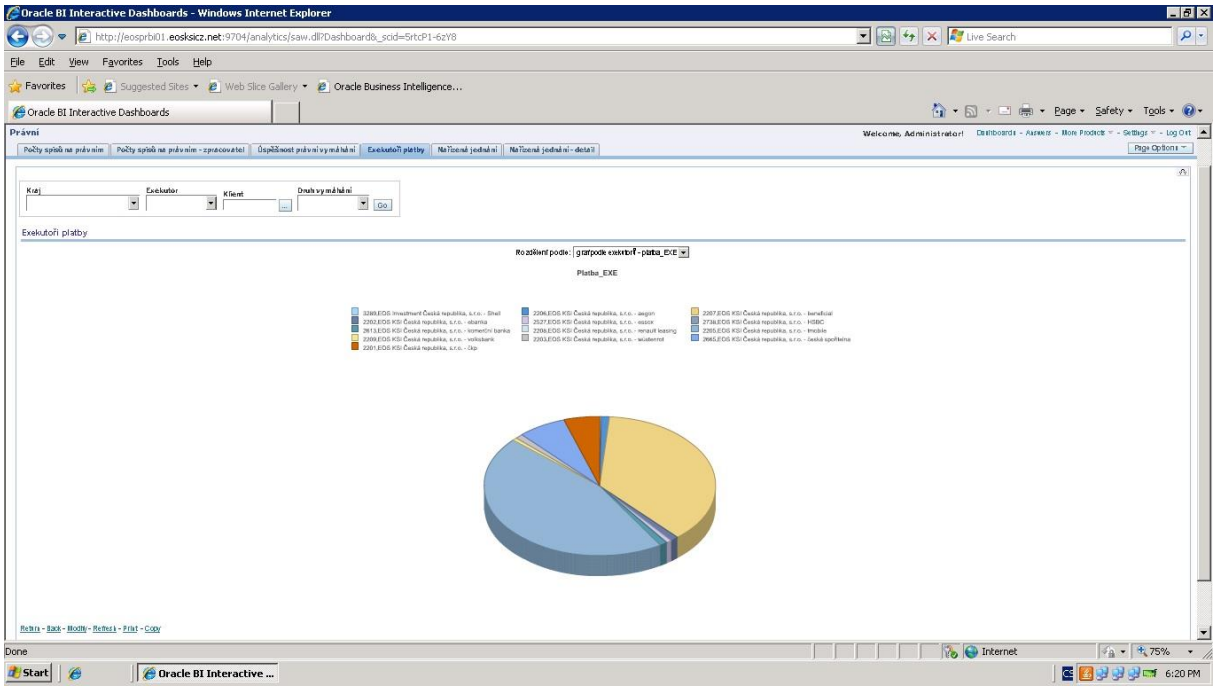
Příloha 6.20 – Zobrazení reportu v Oracle BI Dashboards úroveň Exekutor



Příloha 6.21 – Drill-down na úroveň Druh vymáhání



Příloha 6.22 – Drill-down na úroveň Klient



Příloha 6.23 – Skript na kontroly záznamů v jednotlivých tabulkách, část 1

```
Declare
v_tab_src varchar2(50);
v tab tgt varchar2(50);
n count src number;
n_count_tgt number;
begin
  v_tab_src:= 'KOLLECTO.LEGAL_INFORMATION_TYPE';
  v_tab_tgt:= 'BI.BI_LEGAL_INFORMATION_TYPE';

  select count(1) into n count src
  from kollecto.legal_information_type@KOLLECTO;

  select count(1) into n_count_tgt
  from bi.bi_legal_information_type@KOLLECTO;

  if nvl(n_count_src,0) <>  nvl(n_count_tgt,0)
    then
      dbms_output.put_line ('V tabulkách '||v_tab_src||' a '||v_tab_tgt||'
                             nesouhlasí počet záznamů!');
    else
      dbms_output.put_line ('Plnění z '||v_tab_src||' do '||v_tab_tgt||' je
                             OK!');
  end if;


  v tab src:= 'BI.BI LEGAL INFORMATION TYPE';
  v_tab_tgt:= 'STG0.LEGAL_INFORMATION_TYPE';

  select count(1) into n_count_src
  from bi.bi_legal_information_type@KOLLECTO;

  select count(1) into n count tgt
  from stg0.legal_information_type;

  if nvl(n_count_src,0) <>  nvl(n_count_tgt,0)
    then
      dbms output.put line ('V tabulkách '||v tab src||' a '||v_tab_tgt||'
                             nesouhlasí počet záznamů!');
    else
      dbms_output.put_line ('Plnění z '||v_tab_src||' do '||v_tab_tgt||' je
                            OK!');
  end if;


  v_tab_src:= 'STG0.LEGAL_INFORMATION_TYPE';
  v_tab_tgt:= 'STG1.BI_EXEKUTORI_PLATBY';

select sum(nvl(lpi.others amount,0)) into n_count_src
from legal proposal info lpi
where lpi.legal_information_type_id = 138
and lpi.legal_ce_proposal_id in(
                          select get_exe_navrh_podani(ce.ceid,1)
                          from collection entities ce, clients cl
                          where nvl(ce.status, 'Y') like 'EXNE%'
                          and cl.client id = ce.client id
             and trunc(ce.creation_date)>  to_date('31.12.2007','dd.mm.yyyy')
                             and not exists(
                                          select 1
                                          from dynamic flags df
                                          where df.ceid = ce.ceid
                                          and df.flag_id = 1602
                                          )
                             and not exists (
                                           select 1
                                           from bi vyrazena ce cez
                                           where ce.ceid = cez.ceid
                                          )
                              );
```

Příloha 6.24 – Skript na kontroly záznamů v jednotlivých tabulkách, část 2

```
        select sum(castka exe) into n count_tgt
          from stg1.bi_exekutori_platby;


          if nvl(n_count_src,0) <>  nvl(n_count_tgt,0)
            then
              dbms output.put line ('V tabulkách '||v tab src||' a '||v_tab_tgt||'
                                    nesouhlasí suma CASTKA_EXE!');
            else
              dbms_output.put_line ('Plnění z '||v_tab_src||' do '||v_tab_tgt||' je
                                    OK!');
          end if;

          v tab src:= 'STG1.BI EXEKUTORI PLATBY';
          v_tab_tgt:= 'STG2.P_INS_F_EXEKUTORI_PLATBY';

          select sum(castka_exe) into n_count_src
          from stg1.bi_exekutori_platby;

          select sum(castka_exe) into n_count_tgt
          from stg2.p_ins_f_exekutori_platby;

          if nvl(n_count_src,0) <>  nvl(n_count_tgt,0)
            then
              dbms output.put line ('V tabulkách '||v tab src||' a '||v_tab_tgt||'
                                    nesouhlasí suma CASTKA_EXE!');
            else
              dbms_output.put_line ('Plnění z '||v_tab_src||' do '||v_tab_tgt||' je
                                    OK!');
          end if;

        exception
          when others then  dbms_output.put_line ('Chyba při kontrole '||v_tab_src||'
-> '||v_tab_tgt||' !');
        end;
```