Česká zemědělská univerzita v Praze

Provozně ekonomická fakulta

Katedra informačního inženýrství

Bakalářská práce

Integrace dat v Business Intelligence

Roman Reiter

ČESKÁ ZEMĚDĚLSKÁ UNIVERZITA V PRAZE

Provozně ekonomická fakulta

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

Roman Reiter

Informatika

Název práce

Integrace dat v Business Intelligence

Název anglicky

Data Integration in Business Intelligence

Cíle práce

Téma bakalářské práce je zaměřeno na integraci dat v Business Inteligence. Hlavním cílem bakalářské práce je návrh procesu integrace dat z datového uložiště do konsolidovaného datového skladu v podnikové praxi. Dílčí cíle bakalářské práce jsou:

- Vytvořit kritickou literární rešerši k problematice Business Intelligence.
- Identifikovat nové trendy v oblasti Business Intelligence.
- Provést deskripci nástrojů pro integraci dat.
- Syntetizovat výsledky práce, formulovat závěry a doporučení pro podnikovou praxi.

Metodika

Metodika řešení dané problematiky se skládá ze studia a analýzy odborných publikací, informačních zdrojů a dokumentace. Ve vlastní práci je navržen proces integrace dat prostřednictvím notace BPMN. Na základě syntézy teoretických poznatků a výsledků vlastní práce bude formulován závěr bakalářské práce.

Doporučený rozsah práce

30-40 stran

Klíčová slova

Business Inteligence, datová integrace, znalostní modul, ETL (extract, transform, load)

Doporučené zdroje informací

Fusion Middleware Knowledge Module Developer's Guide for Oracle Data Integrator [online]. 2017, dostupné z: <https://docs.oracle.com/cd/E17904_01/integrate.1111/e12645/toc.htm>

GÁLA, L. – POUR, J. – ČESKÁ SPOLEČNOST PRO SYSTÉMOVOU INTEGRACI, – ŠEDIVÁ, Z. *Podniková informatika.* Praha: Grada, 2009. ISBN 978-80-247-2615-1.

KIMBALL, Ralph; ROSS, Margy; THORNTHWAITTE, Warren; MUNDY, Joe; BECKER, Bob. The Data Warehouse Toolkit 2nd Edition: . Practical Techniques for Building Data Warehouse and Business Inteligence Systems. Wiley Publishing. Canada. 2008. 672 s. ISBN-13: 978-0-470-14977-5

NOVOTNÝ Ota, POUR Jan a MARYŠKA Miloš. Business intelligence v podnikové praxi. Praha: Professional Publishing, 2012. ISBN 978-80-7431-065-2.

NOVOTNÝ Ota, POUR Jan a SLÁNSKÝ David. Business Intelligence: Jak využít bohatství ve vašich datech. Praha: Grada, 2005. ISBN 80-247-1094-3.

Předběžný termín obhajoby

2019/20 ZS – PEF (únor 2020)

Vedoucí práce

Ing. Jan Tyrychtr, Ph.D.

Garantující pracoviště

Katedra informačního inženýrství

Elektronicky schváleno dne 24. 1. 2019

Ing. Martin Pelikán, Ph.D.

Vedoucí katedry

Elektronicky schváleno dne 24. 1. 2019

Ing. Martin Pelikán, Ph.D.

Děkan

V Praze dne 15. 11. 2019

Čestné prohlášení

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

V Praze dne 30.11.2019 _____

Poděkování

Rád(a) bych touto cestou poděkoval(a) Ing. Janu Tyrychtrovi, Ph.D. za odborné vedení a za cenné rady při konzultacích během tvorby této bakalářské práce

Integrace dat v Business Intelligence

Abstrakt

V první části literární rešerše této práce je popsán a definován pojem Business Intelligence, vývoj BI na světovém a českém trhu. Tato kapitola se dále zaměřuje na jednotlivé komponenty, které tvoří architekturu BI. Definuje zdrojové systémy, datový sklad s jeho vrstvami, ETL procesem potřebným pro nahrávání dat a v poslední řade se práce věnuje trendům v oblasti Business Intelligence. V následující kapitole je představen aktuální trend na BI trhu, konkrétně metodika Data Vault pro modelování a vývoj datových skladů, dále jsou identifikovány výhody a nevýhody plynoucí pro organizace ze samotné implementace metodiky. Závěrečná kapitola literární rešerše se zabývá metodikou BPMN pro modelování byznys procesů a popisem grafických prvků využívaných v modelech. V první kapitole praktické části, autor práce představuje postavení jednotlivých zástupců BI řešení na světovém a českém trhu dle společnosti Gartner. Dále byla v praktické části implementována metodika Data Vault na modelu, volně dostupné datové sady, popsán ETL nástroj Oracle Data Integrator, pomocí, něhož byl následně představen způsob integrace dat. V poslední části vlastní práce je představen návrh integrace dat v podnikové praxi.

Klíčová slova: Business Inteligence, Data Vault, znalostní modul, ETL (extract, transform, load), Data Warehouse (DWH), Business Process Model and Notation (BPMN), Oracle Data Integrator (ODI)

Data Integration in Business Intelligence

Abstract

The first part of the literature review describes and defines the concept of Business Intelligence, BI development in the world and Czech markets. This chapter also focuses on the individual components that compose the BI architecture. It defines the source systems from which it is drawn, the data warehouse with its layers, the ETL process needed for data uploading. The thesis deals with trends in the field of Business Intelligence. The next chapter presents the current trend on the BI market, specifically Data Vault methodology for data warehouse modeling, its development and identifies the advantages and disadvantages of organizations from its implementation. The final chapter of the literature review deals with the BPMN methodology for modeling business business processes and describes the graphical elements used in models. In the first chapter, the author presents the position of individual representatives of BI solutions in the world and on theCzech market according to the Gartner. Furthermore, in the practical part, the Data Vault methodology was implemented as the model of freely available data set. The last part of the thesis presents a proposal of data integration in business practice.

Keywords: Business Inteligence, Data Vault, Knowledge Module, ETL (extract, transform, load), Data Warehouse, Business Process Model and Notation, Oracle Data Integrator

Obsah

1 Úvod 11

2 Cíl práce a metodika 12

2.1 Cíl práce 12

2.2 Metodika 12

3 Teoretická východiska 13

3.1 Business Intelligence 13

3.1.1 Vývoj Business Intelligence 13

3.1.2 Komponenty BI řešení 14

3.1.2.1 Zdrojové systémy 14

3.1.2.2 Datové sklady 15

3.1.2.3 ETL 16

3.1.3 Trendy v Business Intelligence 17

3.1.3.1 Agilní metody vývoje datových skladů 17

3.1.3.2 Big Data 18

3.1.3.3 Data Vault metodika pro modelování datových skladů 18

3.2 Data Vault 18

3.2.1 Principy metodiky a výhody pro budování datového skladu 19

3.2.1.1 Integrovanost 19

3.2.1.2 Historizace 19

3.2.1.3 Agilní vývoj 19

3.2.2 Principy a pravidla modelovaní 20

3.2.2.1 Tabulka typu Hub 21

3.2.2.2 Tabulka typu Link 21

3.2.2.3 Tabulka typu Satellite 22

3.2.3 Výhody ETL procesů s metodikou Data Vault 22

3.3 Business Process Model and Notation 23

3.3.1 Sadu grafických prvků 23

3.3.1.1 Tokové objekty 24

3.3.1.2 Plavecké bazény 25

3.3.1.3 Artefakty 26

4 Vlastní práce 27

4.1 Charakteristika vývoje světového a českého BI trhu 27

4.1.1 Oracle 27

4.1.2 Microsoft 28
4.1.3 IBM 28
4.1.4 Trh s dodavateli databází dle společnosti Gartner 29
4.1.5 Trh BI řešení dle společnosti Gartner 30
4.2 Implementace Data Vault metodiky 31
4.2.1 Zvolený model 31
4.2.2 Implementace navrhnutých změn 32
4.3 Integrace dat v ODI 36
4.3.1 Práce s ODI 36
4.3.1.1 Designer 37
4.3.1.2 Topology 38
4.3.1.3 Operator 39
4.3.1.4 Security 40
4.3.1.5 Znalostní modul 40
4.3.1.6 Mapování 41
4.3.1.7 Datový tok 41
4.3.1.8 Změna zaznamenaných dat CDC 41
4.3.1.9 Balíček 41
4.3.1.10 Scénář 42
4.4 Vlastní práce s ODI 42
4.4.1 Vytvoření tabulky Hub 42
4.4.2 Vytvoření tabulky Link 44
4.4.3 Vytvoření tabulky Satelit 45
4.5 Návrh jednotlivých fází integrace dat 47
5 Výsledky 49
5.1 Zhodnocení výsledků 49
6 Závěr 50
7 Seznam použitých zdrojů 51
8 Přílohy 54

Seznam obrázků

Obrázek 1: vrstvy architektury datového skladu [4] 16
Obrázek 2: ETL proces [4] 16
Obrázek 3: Byznysová entita zákazníka a její okolí [7] 21
Obrázek 4: Počáteční a konečná událost [16] 24
Obrázek 5: Grafický prvek [20] 24
Obrázek 6: Grafické prvky brán [20] 25
Obrázek 7: Spojovací grafické [20] 25
Obrázek 8: Plavecké dráhy [20] 26
Obrázek 9: Artefakty [20] 26
Obrázek 10: Magic Quadrant for Analytics [17] 29
Obrázek 11: Magic Quadrant for Data Integration Tools [18] 30
Obrázek 12: Oracle HR Schema [22] 32
Obrázek 13: Tabulka typu HUB (vlastní zpracování) 33
Obrázek 14: Tabulka typu Link (vlastní zpracování) 33
Obrázek 15: Tabulka typu Satelit (vlastní zpracování) 34
Obrázek 16: Data Vault HR schema (vlastní zpracování) 35
Obrázek 17: ODI obrazovky (vlastní zpracování) 37
Obrázek 18: ODI běh procesů (vlastní zpracování) 39
Obrázek 19: ODI změna zaznamenaných dat (vlastní zpracování) 41
Obrázek 20: ODI balíček [13] 42
Obrázek 21: ODI mapování tabulky Hub (vlastní zpracování) 43
Obrázek 22: ODI mapování tabulky typu Link (vlastní zpracování) 45
Obrázek 23: ODI mapování tabulky typu Satelit (vlastní zpracování) 46
Obrázek 24: BPMN návrh procesu integrace (vlastní zpracování) 48

Seznam použitých zkratek

BI	Business Intelligence
DWH	Data Warehouse
ODI	Oracle Data Integrator
ETL	Extract-Transform-Load
ELT	Extract-Load-Transform
LKM	Loading Knowledge Module
IKM	Integration Knowledge Module
BPMN	Business Process Model and Notation

1 Úvod

Tato práce se zabývá problematikou Business Intelligence, využívanou zejména v podnikové praxi velkých organizací, za účelem podpory podnikových rozhodnutí, zlepšení a zkvalitnění procesů uvnitř organizace a snížení časových a finančních nákladů organizace.

Business Intelligence (BI) zastřešuje oblast spadající do světa informačních technologií a díky pokročilým datovým analytickým technikám, umožňuje organizacím ucelený pohled do vnitra organizace. Množství dat celosvětově roste a potřeba ukládání dat rychlým a efektivním způsobem ve vhodné formě pro operativní a analytické zpracování, se zvyšuje. Tato potřeba je klíčovou vlastností pro vedení úspěšné organizace.

Základní podstatou BI je přeměnit data v organizaci na informace, odhalit vazby mezi informacemi a proměnit je v znalost, potřebnou k rozhodování nejvyššího managementu k dosahování jejich cílů v obchodní činnosti.

Smyslem praktické části je představit trendy využívané v integraci dat za poslední období, konkrétně metodiku modelování Data Vault, dále představit konvenční integrační nástroje využívané zejména ve velkých korporacích a na zvoleném nástroji ukázat možný způsob datové integrace dle zvolené metodiky. Poslední část praktické části se zabývá návrhem způsobu integrace dat pomoci metodiky BPMN a doporučením pro organizace.

2 Cíl práce a metodika

2.1 Cíl práce

Téma Bakalářské práce je zaměřeno na integraci dat v Business Intelligence. Hlavním cílem práce je návrh procesu integrace dat z datového uložiště do konsolidovaného datového skladu v podnikové praxi. Dílčí cíle bakalářské práce jsou:

- Vytvořit kritickou literární rešerši k problematice Business Intelligence.
- Identifikovat nové trendy v oblasti Business Intelligence.
- Provést deskripci nástroje pro integraci dat.
- Syntetizovat výsledky práce, formulovat závěry a doporučení pro podnikovou praxi.

2.2 Metodika

Metodika řešení dané problematiky se skládá ze studia a analýzy odborných publikací, informačních zdrojů a dokumentace. Z těchto zdrojů jsou zapracované klíčové oblasti Business Intelligence, včetně současného stavu trhu s BI platformami.[1][2][3]

Pro účel vlastní práce byl vybrán nástroj: Oracle Data Integrator. Výběr zmíněného nástroje byl proveden na základě studia odborných publikací a na postavění zástupců BI řešení na světovém trhu v roce 2019 dle magického kvadrantu od společnosti Gartner. Řešení vlastní práce, ve které je realizována implementace BI řešení je dosaženo znalostí prostřednictvím vybraného ETL nástroje a znalostí databázového systému Oracle SQL Server od společnosti Oracle. Ve své práci jsem využil volně dostupné databázové řešení od společnosti Oracle, a to konkrétně Oracle HR Schema. [4][6][12][13][14][15][17][18]

Hlavním trendem, kterým se autor této práce věnuje ve své odborné praxi je aplikování principů modelování pomoci metodiky Data Vault. Jedná se o zcela zajímavý trend v oblasti BI, který víc získává oblibu na trhu a má snahu dodávat agilní datové sklady, které jsou schopné reagovat rychle, a hlavně efektivně na změny. [6][7][8][9][10][11]

Pro praktickou ukázku jsem aplikoval principy metodiky na výše zmíněném modelu HR Schema od společnosti Oracle.

Dosažení hlavního cíle mé bakalářské práce je realizováno nastudováním standardu pro modelování business procesů, konkrétně pomoci metodiky Business Process Model and Notation a návrhem procesu integrace dat prostřednictvím notace BPMN. [16][20][21]

Na základě syntézy teoretických poznatků a výsledků vlastní práce bude formulován závěr kapitole výsledků.

3 Teoretická východiska

3.1 Business Intelligence

„*Business Intelligence (BI) je termín, označující celý komplex činností, úloh a technologií, které dnes stále častěji tvoří běžnou součást řízení podniků a jejich informačních systémů*.“ [1]

Pro Business Intelligence se těžko hledá vhodný český termín, který by přesně a úplně vystihl problematiku, kterou se BI zabývá. Původní pokusy o překlad jako podniková inteligence, obchodní inteligence nebo inteligentní obchod a další pokusy přeložit termín se jevili jako nevhodné, a proto autoři raději zůstali u anglického termínu.

3.1.1 Vývoj Business Intelligence

Prvotní pokusy směřující k podpoře manažérských a analytických úloh v podnikovém řízení se začali objevovat již v průběhu 70. let minulého století. V první polovině 80. let se začali publikovat relevantní vědecké práce – například *CEO goes On-Line of prof. Rockart* a později v druhé polovině 80. let přišli na americký trh firmy Comshare a Pilot, které představili řešení pro společnosti s komerčními produkty, založenými na multidimenzionálním uložení a zpracování dat, označovanými jako EIS (Executive Information System). Trh s EIS produkty začal velmi rychle rozvíjet a o tyto produkty vzniknul na začátku 90. let zájem i na IS/ICT českém trhu.

Další trendy v oblasti BI a multidimenzionálních technologiích, které se začali v USA objevovat v 80. a začátkem 90. let jsou datové sklady (Data Warehouse) a datové tržiště (Data Marts). Za rozvojem těchto technologií stáli Ralph Kimball a Bill Inmon. Na českém trhu se uplatnění datových skladů a tržišť začalo uplatňovat až v druhé polovině 90. let. V důsledku narůstajícího objemu dat ve souvislosti datových skladů se v průběhu 90. let začali prosazovat technologie a nástroje určené pro tzv. dotazování dat (Data

Mining). Tyto nástroje jsou založené na propracovaných analýzách dat, které využívají nejrůznější matematické a statistické metody. [1][2]

3.1.2 Komponenty BI řešení

„Na úvod je nutné zdůraznit, že konkrétní uspořádání jednotlivých komponent v řešení BI se může výrazně měnit podle situace a potřeb zákazníka nebo podniku.“ [1]

Po dlouhé době vývoje různých BI řešení podle potřeb zákazníků a podniků, se obecná koncepce řešení BI architektury ustálila, ale díky různým problémům řešených pomoci BI nástrojů vede k tomu, že architektura má několik vývojových větví a samotná implementace komponentů se můžu podstatně lišit.

3.1.2.1 Zdrojové systémy

Zdrojové systémy, někdy označované jako primární nebo transakční systémy, jsou důležité pro podporu aplikací BI. Společnou vlastnosti zdrojových systémů je architektura podporující ukládání a modifikaci dat v reálnem čase, nejsou však navržena pro analytické úlohy. Data můžeme získávat ze dvou základních typů zdrojových systémů, a to z interního nebo externího zdrojového systému. Interní zdrojové systémy dodávají data v rámci organizace příkladem může být CRM (Customer Relationship Management) nebo ERP (Enterprise Resource Planning). Data z těchto zdrojů jsou již obohacena o cenné informace. Za externí zdroje považujeme všechny zdroje mimo organizaci, které můžou být zdrojem BI řešení, jedná se například o databáze podnikatelských subjektů, výstupy statistických úřadů apod.

Zdrojové systémy jsou často jediným vstupem pro BI řešení. Je nutné u všech BI řešení zajistit analýzu těchto zdrojů z pohledu potřeb řízení organizace, identifikovat relevantní data vhodné pro řízení a v posledním kroku vzájemnou integraci. Právě integrace představuje nezbytný předpoklad funkčních BI řešení, která je pracovně a finančně náročnější. [1][3]

3.1.2.2 Datové sklady

Datový sklad představuje nejvýznamnější pilíř v rozvoji podnikových informačních systémů. *„Datový sklad (Data Warehouse) je integrovaný, subjektivně orientovaný, stály, a časově rozlišený souhrn dat, uspořádaný pro podporu potřeb managementu“* [1]

Interpretace pojmů definice:

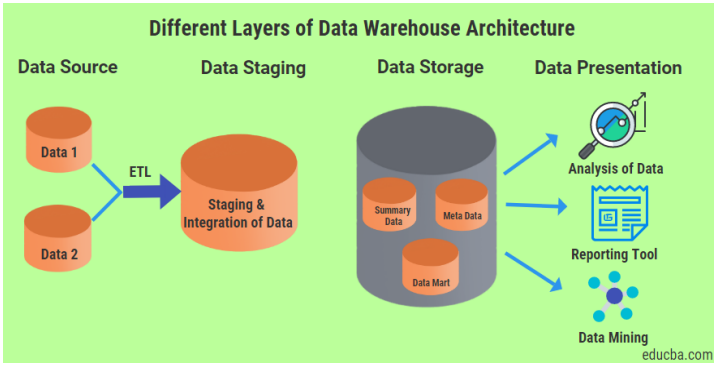
- **Subjektově orientovaný** – lze definovat jako data rozdělena podle jejich typu, nikoli podle aplikaci, z kterých pocházejí.
- **Integrovaný** – data se ukládají v rámci celého podniku, a ne pouze pro jednotlivé oddělení.
- **Stály** – všechny data jsou přístupná v datovém skladu a nevznikají ručním pořízením a nelze je ani měnit pomoci uživatelských nástrojů. Data jsou načítána ze zdrojových systémů a existují po celou dobu životnosti datového skladu.
- **Časově rozlišený –** důležitý aspekt fungujícího datového skladu, protože potřebujeme znát historii, jak byli data načtena, aby bylo možné provádět analýzy za určité období.

3.1.2.2.1 Vrstvy datového skladu

Při vývoji DWH se často vyskytuje potřeba rozdělit sklad do třech základních vrstev/úrovní: [3][4]

- **Vstupní vrstva** – slouží jako dočasné úložiště dat ze zdrojových systémů. Data můžou pocházet z různých externích nebo interních zdrojů a můžou nabývat různé podoby a hlavní účel této vrstvy je sjednotit. Data se ve zdroji nijak netransformují, pouze se přidají technické sloupce, například ID zdrojového systému a časová značka.
- **Business jádro** – slouží pro konsolidaci, historizaci a popis dat. Jádro tvoří hlavní část systému a zaručuje jednotný pohled na podnikové data, podporuje uložení správné historických dat, sjednotí data do srozumitelné formy v podobě tabulek a sloupců dle datových typů a významu.
- **Reportovací vrstva** – slouží pro podporu analýz koncových uživatelů. V reportovací vrstvě najdeme detailní agregovaná data uložená v integrovaném

dimenzionálním modelu, který je optimalizován pro dotazy koncových uživatelů a podporuje správné uložení historických dat.

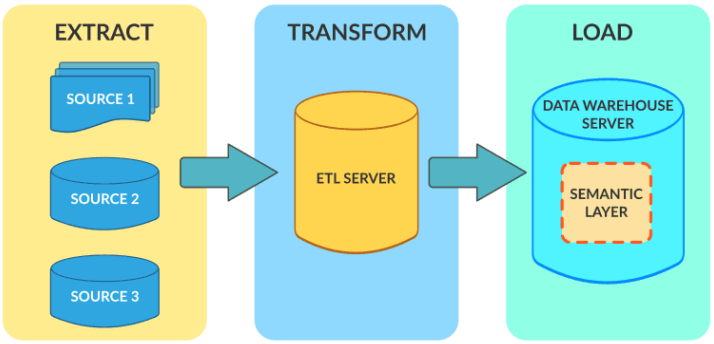


Obrázek 1: vrstvy architektury datového skladu [4]

3.1.2.3 ETL

Extraction, Transofrmation and Loading – ETL je jednou z nejdůležitějších komponent celého BI řešení. ETL lze definovat jako datovou pumpu, jejími dílčími úkolem je získávaní a vybírání dat ze zdrojových systémů (Extraction), úprava a čištění dat do požadované formy (Transformation) a nahrání dat do specifických struktur datového skladu (Loading). [1][3]

„Extract-Transform-Load (ETL) systém, základem datového skladu.“ [4] (překlad autora) Základním aspektem fungujícího DWH je správně navrhnutý ETL systém, který načítá data z různých zdrojových systémů, prosazuje datovou kvalitu s důsledností na standardy, zaručí společné použití dat z rozdílných datových zdrojů a dodá data do reportovací vrstvy pro koncové uživatele.



Obrázek 2: ETL proces [4]

3.1.3 Trendy v Business Intelligence

3.1.3.1 Agilní metody vývoje datových skladů

„Agilní je dynamický, rychlý, interaktivní, přizpůsobiví, zábavný a hravý, rychle reagující na změnu.“ [5] Agilní metoda vývoje je interaktivní způsob řízení projektů, kterého základem je dodávat produkt průběžně v malých iteracích zákazníkovi. Zákazník se aktivně podílí při dodávání produktu a pomáhá zajistit rychlé reagování na změny.

Základním stavebním kamenem je manifest, který definuje následující principy:

- **Jednotlivci a interakce před procesy a nástroji** – klíčovým faktorem je spolupráce týmu před individuálně pracujícími jednotlivci. K dosáhnutí výsledků jim poslouží procesy a nástroje, ale nejsou nijak klíčové pro úspěšný produkt.
- **Fungující software před vyčerpávající dokumentací** – dokumentace je důležitá pro situace a oblastí, které nejsou intuitivní a snadno pochopitelná a neměla by převážit vlastní produkt.
- **Spolupráce se zákazníkem před vyjednáváním o smlouvě** – dalším klíčovým faktorem je komunikace se zákazníkem. Zákazník je vnímán jako součást týmu a měl by se podílet při dodání produktu. Smlouvy jsou důležité, ale nesmí se brát jako prostředek pro nahrazování komunikace se zákazníkem.
- **Reagování na změny před dodržováním plánu** – plány jsou důležitou součástí projektu, můžou být způsobeny novým trendem nebo změnou technologie, ale projektové plány by neměly řídit součinnost spolupracujících firem.

Agilní tým se rozděluji do třech základních rolí:

- **Scrum Master** – hlavním cílem je pomáhat ostatním členům týmu s komunikaci mezi sebou, ale i s ostatními úseky podniku, motivovat tým k lepším výsledkům, ovládat metodiku Scrumu a být zodpovědný za její porozumění a aplikaci.

- **Product Owner** – primárním cílem Product Ownera je porozumění produktu a nese primární odpovědnost za projekt. Jedná se o roli, která zastupuje zákazníka a

stará se o to, aby požadavky a představy ohledně produktu byly splněné, přitom Product Owner může být zaměstnancem klientské firmy nebo najatý externista.

- **Self-organized tým** – jedná se o tým lidí, kteří jsou samo organizovatelný a mohou se mezi sebou zastoupit. Cílem týmu je dodat část funkčního produktu na konci iterace.

3.1.3.2 Big Data

Strategie Big Data je se zaobírá se zpracováním velkého množství nestrukturovaných dat, jejichž velikost není schopné zpracovávat běžně používanými softwarovými nástroji v rozumném čase. Nestrukturovaná data můžeme rozdělit na dva typy, jedním jsou nestrukturovaná opakující se data a druhým typem jsou nestrukturovaná neopakující se data. Při nestrukturovaných opakujících dat, se klade důraz na přístup, monitorování, zobrazení, analýzu a vizualizaci. Při nestrukturovaných neopakujících datech se zcela téměř soustředí na odstranění textových nejasností. Nejznámější nástroj pro zpracování velkých dat je *open source* (volně dostupný) software Hadoop. [6]

3.1.3.3 Data Vault metodika pro modelování datových skladů

Architektura Data Vault nabízí jedinečné řešení pro businessové a technické problémy, její úsilí je zaměřeno na integraci dat v celém podniku a je postavena na solidních, základních konceptech. Hlavním klíčem pro porozumění Data Vaultu, je porozumění byznysového konceptu, jakmile jsou byznysové potřeby podniku zmapovány a odborník má pevnou a přesnou představu o tom, jak podnik funguje, může se pak zahájit budování DWH. [7]

Podrobněji popíšu principy a pravidla pro modelování v samostatné kapitole.

3.2 Data Vault

Metodika Data Vault, konkrétně její verze 2.0, byla poprvé představena v roku 2013 a její původní verze sahá až do roku 1990 a poprvé byla představena v roku 2000. Za zakladatele metodiky je považován Dan Linstedt.

3.2.1 Principy metodiky a výhody pro budování datového skladu

„Metodika Data Vault je velmi efektivní pro modelování datových skladů, protože je optimalizována pro integraci, historizaci a splňuje požadavky pro agilní vývoj.“ [8] (překlad autora). Výhody modelování pomoci metodiky jsou realizovatelné až tehdy, pokud se důsledně uplatňují modelovací vzory, nadefinované v metodice Data Vault.

3.2.1.1 Integrovanost

„Datová integrace znamená: Shromažďování dat z různých zdrojů a integrace daných dat strukturovaným a smysluplným způsobem.“ [8] (překlad autora) Modelování DWH vyžaduje specifický požadavky, protože se primárně zabývá s nahráváním dat z různých datových zdrojů v různých časových úsecích. V rámci integrace se buduje DWH tak, že na začátku máme data z různých na sobě nezávislých zdrojových systémů a na jejím konci máme data sjednocená napříč celým podnikem.

Sjednocená data na jednom místě nám poskytnou nízkou redundanci dat a nejvyšší stupeň integrace. Docílí se to tak, že do jedné sdílené tabulky můžou nahrávat data v rámci dané entity ze všech zdrojových systémů spolu se záznamem o jejich fyzickém uložení – ID zdrojového systému a pomoci ETL procesu, logických a byznysových pravidel se můžou data různě duplikovat a upravovat do výsledného sjednoceného pohledu.

3.2.1.2 Historizace

Historizace je jedna z nejdůležitějších vlastností každého datového skladu. Podstatnou vlastností metodiky Data Vault je tzv. „insert only strategy“, který se využívá při plnění Satelitu a jedinou povolenou DML operací je insert statement. Tím se docílí, že v jádře datového skladu budou všechny nová nebo změněná data a historii můžeme sledovat pomoci časové značky, kdy se záznam nahrál. [10]

3.2.1.3 Agilní vývoj

Základní principy agilního vývoje jsem představil v kapitole 3.1.3.3 a v této sekci se zaměřím na využití agilního způsobu vývoje při použití metodiky Data Vault.

Data Vault metodika 2.0 není jenom o modelování, ale zahrnuje taky osvědčené postupy pro architekturu, nahrávání dat a metodiku. Data Vault model se vytváří od úplných základů, je připravena na jakékoli změny v průběhu implementace a je přímo

přizpůsobená pro agilní přístup. Důsledkem je fakt, že požadavky zákazníka se v průběhu implementace můžou často měnit. V přístupu Data Vault jsou tři klíčové funkce, které nám použít, je jako agilní techniku modelování: [11]

- Flexibilita
- Produktivita
- Škálovatelnost

S agilním způsobem vývoje a dodáváním části produktů v rámci dvou až tří týdenních iteracích, nám pomůže identifikovat chyby hned na začátku a pružně na ně reagovat

3.2.2 Principy a pravidla modelovaní

Metodika Data Vault vznikla hlavně pro modelování DWH a pro uložení jednotlivých entit podniku slouží následující konstrukty:

• Hub

• Link

• Satellite

Při dodržení konceptu konstruktorů, Data Vault zajišťuje efektivní modelování všech logických a byznysových rozhodnutí, při tvorbě datového skladu. Jednotlivé konstrukty obsahují několik společných technických atributů. [7][9]

SurrogateKey (SK) – ID generován ze sekvence databází

HashKey (HK) – ID entity transformované pomoci hash funkci

LoadDatetime – časová značka, kdy se záznam nahrál

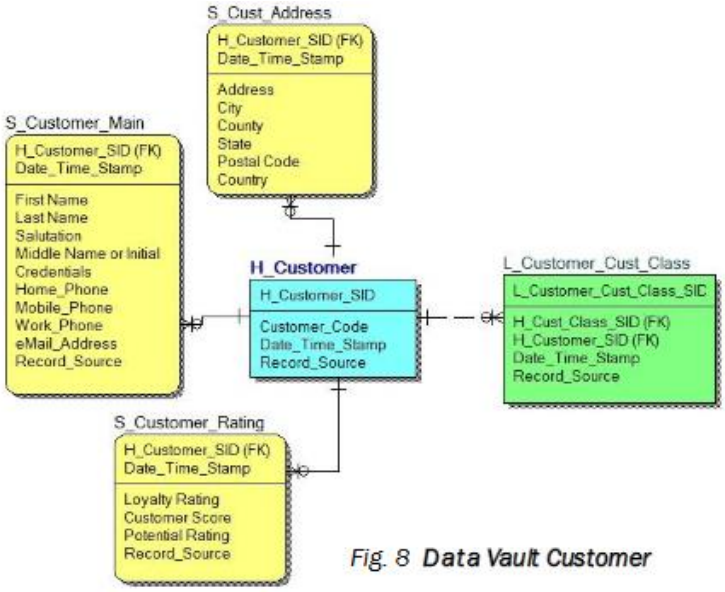
SourceSystem – ID zdrojového systému, informace o původně záznamu

HashDiff – agregovaný obsah atributů transformován pomoci hash funkce

Jednou z efektivních metod spájení tabulek je, využití hash funkce např. SHA1, MD5. Pokud je business klíč kompozitní, tak se musí všechny položky zahrnout do výpočtu HashKey. [9] Jako příklad uvedu, že chceme v tabulce *Account* (účty) vytvořit unikátní business klíč, tak musíme zkombinovat číslo účtu a kód banky abychom dosáhli unikátnosti. Výhodou využití hash funkce je efektivnější dotazování mezi tabulkami.

Jedním z důvodů je odstranění závislosti při načtu datového skladu, protože HashKey se vypočítá už při extrakci dat ze zdrojového systému a pak můžou být nezávisle

načteny závislé entity (Satellite, Link) v nezávislém pořadí, protože primární klíč je dopředu známe a nemusíme čekat na jeho vygenerovaní. HashKey nahrazuje sekvenčně generovaný SurrogateKey a existují zastánci jedné nebo druhé metody. [9] Nicméně v komplexnějších podnicích můžeme narazit, kde se využívají oba přístupy.



Obrázek 3: Byznysová entita zákazníka a její okolí [7]

3.2.2.1 Tabulka typu Hub

Tabulka typu Hub reprezentuje hlavní byznysovou entitu, jako například zákazník, prodejce, produkt atd. Tabulka Hub je formovaná pro ukládání přirozeného *Natural Business Key* (představuje přirozený klíč, definující hlavní byznysový identifikátor entity, např. účet klienta) a záznam se uloží jenom při prvním příchodu entity daného Natural Business Key, neobsahuje žádné popisní informace a cizí klíče. Kardinalita vztahu tabulky Hub musí být 1:1 z další entity byznysového konceptu. Krom Natural Business Key se do tabulky ukládají ještě technický atributy jako ID sekvence, Load Datetime, Source System ID.

Je možné oba atributy Load Datetime a Source System vynechat, protože ani jeden není součástí žádného klíče. Technicky to neovlivní fungování a smysl tabulky, jenom se přijde o informaci, kdy se poprvé záznam nahrál do tabulky a odkud se vzal. [8]

3.2.2.2 Tabulka typu Link

Tabulka typu Link je vazební konstruktor, který spájí 2-n byznysové entity mezi sebou. Každý Link je založený na unikátní a specifický vazbě Natural Business Key.

Z tohohle pohledu se Link podobá Hubu, protože stejně jako Hub, link neobsahuje žádné popisní informace, stejně jako u Hub konstruktoru se do tabulky nahrává záznam jenom při prvním příchodu entity, navíc Link neobsahuje vlastní Natural Business Key ale složený klíč dvou nebo více entit. Stejně jako u Hub tabulky, Link obsahuje technické atributy jako ID sekvence, Load Datetime, Source System ID.

„V Data Vault metodice existuje jenom many-to-many (N:M) relace.“ [8] (překlad autora) Metodika Data Vault implicitně stanovuje kardinalitu vztahů mezi byznys entitami jako N:M, protože z modelovací perspektivy se modelář, vývojář, architekt můžou víc soustředit na identifikování vazby mezi Natural Business Key a míň na určení kardinality vazby. Nevýhodou vazby N:M, je že na první pohled není jasná skutečná kardinalita, další nevýhodou je pokušení modelovat Link jako samostatnou entitu. [9]

3.2.2.3 Tabulka typu Satellite

„Satellite zachytává všechen kontext po celou dobu historie v datovém skladu.“ [8] (překlad autora) Z pohledu Data Vault metodiky je satelit nejtěžší pracovní konstrukt pro modelování, obsahuje všechna popisná data spolu pro business koncept a relace. Vyplývá to z faktu, že Hub může mít na sobě navázaných víc satelitů dle významu atributů, které obsahuje a je jenom na modelářovi, vývojáři nebo architektovi, jak satelity zhotoví pro danou byznys oblast.

Kardinalita vztahů satelitů je 1:N a jako jediná zachytává historii, protože jako jediný konstrukt v Data Vault metodice, který využívá Load Datetime – časovou značku, jako součást Natural Business Key. Z tohoto pohledu dokáže satelit, zachytávat změny jako pomalu měnící se dimenze typu 2.

3.2.3 Výhody ETL procesů s metodikou Data Vault

„V první řadě je třeba říct, že každá sebedokonalejší věc se hodí pro určité scénáře a pro některé se nehodí vůbec. To platí i o Data Vault 2.0.“ [9] Mezi hlavní výhodu modelování pomoci metodiky Data Vault oproti klasickému dimenzionálnímu, je jeho různorodost entit a přizpůsobení k dnešnímu agilnímu přístupu tvorby datového skladu. Velikou výhodou metodiky, je že při dodávkách po malých částech, je zákazník schopen nahlédnou a sáhnout si na řešení a za včas reagovat na případné změny. U dimenzionálního modelu je to obtížnější, protože každý zásah do modelu znamená, pracnou úpravu současných objektů v databázi a pracnou úpravu historie nahrávaných dat.

..

Kdy uvažovat o Data Vault způsobu modelování:

- Velikost DWH bude narůstat v čase a bude obsahovat mnoho dalších datových zdrojů
- DWH je za potřeby budovat iterativně za pomoci agilní metody vývoje
- DWH se bude v dlouhém časovém horizontu často rozvíjet dle požadavkům na změnu a jeho samotný vývoj bude stále pokračující záležitosti

Kdy se využití Data Vault způsobu modelování nehodí:

- DWH se bude skládat z malého počtu entit (řádově do 20entit) a z malého počtu zdrojových systémů
- DWH má naplánováno jenom malou přesně vymezenou část business potřeb a nebude potřeba ho dále významně rozvíjet
- DWH slouží pro ukázku reportingových nástrojů jako Proof of concept

Implementace Data Vault metodiky se spíše hodí pro velká řešení v často se měnícím prostředí. [9]

3.3 Business Process Model and Notation

BPMN (Business Process Model and Notation) se stal de-facto standardem pro modelování byznys procesů. Je navržen tak, aby jej mohli využívat různí účastníci projektu, kteří navrhují, spravují nebo mění business procesy, ale zároveň byl dostatečně precizní, aby umožňoval překlad BPMN notace pro softwarové komponenty. BPMN je jednoduchá notace, připomínající vývojový diagram a je zároveň nezávislá na konkrétním prostředí implementace. [16]

3.3.1 Sadu grafických prvků

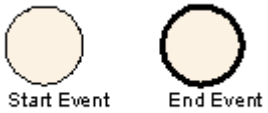
BPMN notace užívá poměrně komplexní sadu grafických prvků, potřebných pro modelování podnikových procesů, z kterých vznikají jednoduché diagramy, pomoci, nichž je pro vlastníky procesů, vývojáře a analytiky snazší porozumět procesním tokům i samotným procesům. BPMN se rozděluje na čtyři základní kategorie: tokové objekty, spojovací objekty, plavecké dráhy a artefakty.[16][20]

3.3.1.1 Tokové objekty

Mezi tokové objekty patří:

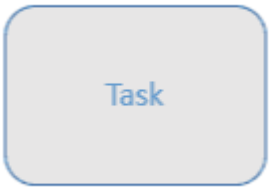
Události jsou značené kolečkem a představují dej, který má přímý vliv na chod podnikových procesů. Mezi hlavní typy událostí patří:[16][20]

- **Počáteční událost** – je zobrazena jednoduchým kolečkem a představuje spouštěč procesů
- **Konečná událost** – je značená kolečkem s tlustým okrajem a představuje výsledek aktivity či celého procesu



Obrázek 4: Počáteční a konečná událost [16]

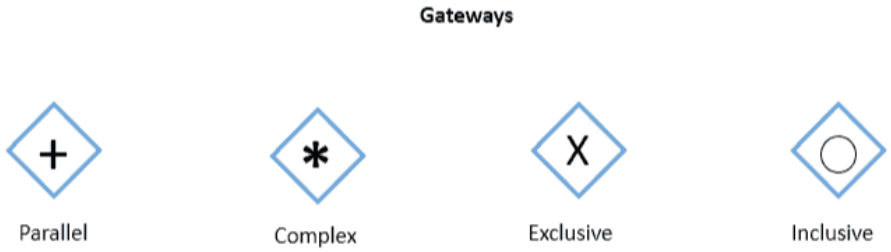
Aktivity jsou prvky, které představují jednotlivé činnosti, odehrávající se ve vnitřku procesu. Zobrazují se pomoci obdélníka so zaoblenými hranami



Obrázek 5: Grafický prvek [20]

Brány jsou prvky umožňující větvení a slučování toků nebo procesů v závislosti na uvedených podmínkách. Znázorňují se pomoci kosočtverce. Brány se rozdělují na několik typů:

- **Exkluzívní** – vytváří několik cest toku procesu, podmínkou je, že tok procesu může projít jenom jednou z možných cest.
- **Inkluzívní** – použití je vhodné na místě, kde tok procesu může projít bránou víc než jednou cestou. Po projedení brány se většinou všechny cesty sloučí do jedné.
- **Komplexní –** možnost využití v případu, když je nemožné použít předcházející druhy brán a kde dochází k dělení cest v několika bránách.
- **Paralelní –** použití v případu, když tok procesu proudí vícero cestami najednou.

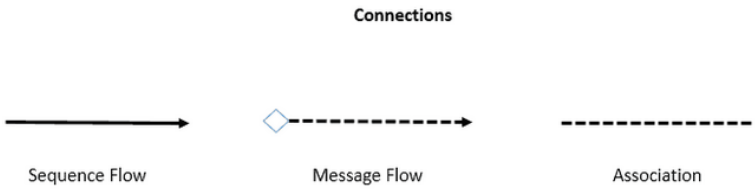


Obrázek 6: Grafické prvky brán [20]

3.3.1.1.1 Spojovací objekty

Spojovací objekty spájejí tokové objekty dohromady a dělí se na tři typy:

- **Sekvenční tok** – znázorňuje postupnost procesních tok. Zdrojem a cílem je vždy aktivita, událost nebo brána. Je znázorněn pomoci plné čáry zakončená šipkou v směre běhu procesu.
- **Tok zpráv –** možnost využití pro komunikaci mezi dvěma nebo vícero bazénama. Tok zpráv se znázorňuje přerušovanou čárou, která má na začátku kolečko a na konci šipku v směre běhu procesu.
- **Asociace –** používá se pro připojení textu nebo artefaktů k tokovým objektům. Je znázorněna přerušovanou čárou.



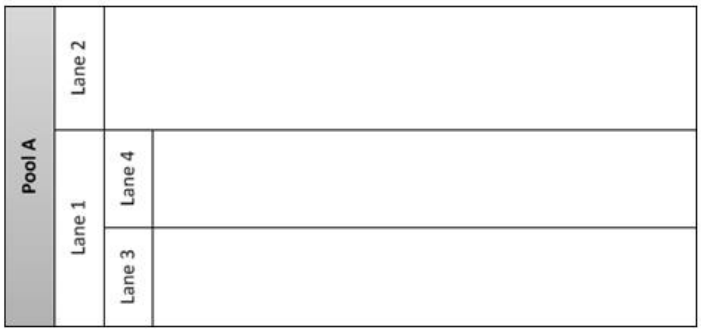
Obrázek 7: Spojovací grafické [20]

3.3.1.2 Plavecké bazény

Plavecké dráhy slouží na organizování a kategorizaci činností. Mezi plavecké dráhy patři dva typy prvků:

Bazény jsou hlavním prvkem procesu a jsou největší částí diagramu a reprezentují účastníky procesu (společnosti, klienti nebo oddělení). Každý Pool obsahuje jeden proces sloužící k organizaci a kategorizaci účastníků procesu a činností v diagramu.

Dráhy se využívají na organizaci a kategorizaci ve vnitřku bazénu na základě funkcí a rolí a můžou být rozděleny na jednotlivé řádky dráhy, které představují paralelní procesy a jejich závislosti mezi účastníky, které jsou takto vizuálně odděleny[16][20].

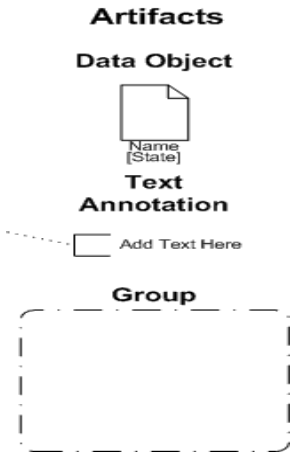


Obrázek 8: Plavecké dráhy [20]

3.3.1.3 Artefakty

Umožnují vývojářům přidat další informace do modelu, díky tomu je pak model čitelnější. Dělí se na tři předdefinované objekty:

- **Datové objekty –** ukazují na data, nevyhnutné pro vykonání daný činnosti
- **Skupiny –** využívají se na seskupení různých aktivit, bez vlivu na samotný tok diagramu.
- **Anotace –** anotace dodávají přehlednost a srozumitelnost celému diagramu



Obrázek 9: Artefakty [20]

4 Vlastní práce

Cílem praktické části bakalářské práce je demonstrovat ukázku využití metodiky Data Vault a pomoci vhodně zvoleného komerčního ETL nástroje, demonstrovat integraci dat na praktickém příkladu.

4.1 Charakteristika vývoje světového a českého BI trhu

Mezi světovou špičku na trhu s BI platformami a aplikacemi, kteří také aktivně působí v České republice patří, Oracle, Microsoft a IBM. Od roku 2007 prošli jednotlivé platformy a aplikace výraznou konsolidací. [12]

4.1.1 Oracle

Oracle začal dominovat na BI trhu po převzatí Hyperion Solutions Corporation v roce 2007. Silnou pozici na trhu měl Oracle ještě před převzatí Hyperionu, kterou si udržel zvlášť díky vlastním vyspělým technologiím rozvíjeným na platformě Oracle Database.

BI platforma od Oracle zajišťuje následující produktové portfolio pro všechny oblasti BI:

- Jednotný způsob administrace a zabezpečení
- Jednotný způsob uložení a správy metadat.
- Integrovaný uživatelský portál
- Vývojové nástroje pro integraci BI aplikací, za podpory webových služeb
- Stará se o komplexní správu dat, včetně sdílení metadat na centralizovaném místě
- Sdílení uživatelských data a sledovat události vybraným skupinám uživatelů podle nadefinovaných pravidel.
- Podpora integrace s Microsoft Office

Hyperion poskytl významné výhody především v oblasti plánování a finančního řízení, čím umožnil pro Oracle BI platformu, vytvořit moderní manažerský informační systém. Oracle BI platforma rovněž disponuje v oblasti správy a integrace dat, potvrzuje to fakt, že

Oracle ve srovnání s konkurencí, měl určitý náskok, protože začal dřív se skupováním dodavatelů BI aplikací a komplementárních produktů. [12]

4.1.2 Microsoft

Pro společnost Microsoft byl významný pokrok, když získal společnost ProClarity v roce 2006. Dle agentury Gartner byl na trhu v roce 2007 Microsoft brán, jako vyzyvatel a o tři roky později se nacházel na pozici lídrů světových dodavatelů BI platforem. Za zmínku stojí fakt, že agentura Gartner umístila ve svém hodnocení společnost Microsoft na pozici lídrů vícekrát než kterémukoli konkurentovi.

BI platforma Microsoftu zajišťuje následující produktové portfolio pro všechny oblasti BI:

- Komponenty potřebné pro integraci a správu dat ze všech interních nebo externích datových zdrojů.
- Jednotný způsob administrace a zabezpečení.
- Jednotný způsob uložení a správy metadat.
- Uživatelský portál propojující všechny nástroje BI.
- Integrační nástroj s podporou webových služeb, pro vývoj a integraci BI řešení.
- Sdílení uživatelských data a sledovat události vybraným skupinám uživatelů podle nadefinovaných pravidel.
- Podpora integrace s Microsoft Office

Microsoft se svojí BI platformou směřuje do širokého segmentu trhu, na němž se nachází organizace různých velikostí a odvětví. Vděčí za to svému marketingovému a obchodnímu přístupu. Pro český trh je podstatní, že Microsoft dokáže vytvořit plnohodnotný manažerský informační systém, při využití vlastních a partnerských nástrojů.[12]

4.1.3 IBM

Posunutí IBM na přední příčky BI trhu, se dostavilo až odkoupením Cognosu, do té doby IBM nedisponovala s žádnou vlastní ucelenou BI platformou. IBM byla schopná poskytovat vlastní obchodní BI řešení na základě databázových a integračních technologií pomoci IBM DB2 DW Edition a IBM Information Server. Gartner v roce 2010 řadí IBM

mezi popřední lídři na trhu, protože Cognos se řadí k nejlepším BI platformám vůbec. [10][17][18]

BI platforma IBM zajišťuje následující produktové portfolio pro všechny oblasti BI:

- Jednotná platforma a schopnost rozsáhlé konverzace dat.
- Využití servisně orientované architektury posilující škálovatelnost, otevřenost a integraci jednotlivých komponent daného řešení pro podnikové informační systémy.
- Schopnost využití a získávaní dat z různých datových zdrojů, relačních databází, textových souborů, excelu, ručně vkládaných hodnot a ze všech komponent aplikace.
- Možnost využití Cognosu na všech platformách.

Propracovaný koncept řízení výkonnosti podniku, který funguje na bázi vyvážené srovnávací tabulce a je jednoduše sledovatelný a srozumitelný pro běžné uživatele.

4.1.4 Trh s dodavateli databází dle společnosti Gartner

Obrázek níže zobrazuje zástupce dodavatelů cloud a klasických relačních databázových řešení dle schopnosti plnit své cíle a podle celistvosti jejich vize do budoucnosti. Kvadrant 2. představuje lídři na trhu, kteří zastupují zejména představitelé klasických databází. Kvadrant 3. zobrazuje zejména představitelé cloudových řešení, kteří jsou v pozadí se svými vizemi a schopnostmi plnit své cíle. [17]

Obrázek 10: Magic Quadrant for Analytics [17]

4.1.5 Trh BI řešení dle společnosti Gartner

Existuje velká škála komerčních a *open source* (volně dostupných) nástrojů pro datovou integraci. Pro vhodný výběr nástrojů jsem na obrázku níže, použil Gartner Magic Quadrant for Data Integration Tools 2019 od společnosti Gartner. Obrázek zobrazuje aktuální stav integračních nástrojů a postavení jednotlivých poskytovatelů BI platforem podle jejich schopnosti plnit své cíle a podle celistvosti jejich vize do budoucnosti. Kvadrant 2. prezentuje vůdčí firmy v oboru, zatímco 3. kvadrant prezentuje zejména specializované firmy, se zaměřením na uspokojení potřeb jednoho specifické zákazníka. Podstatnou roli při hodnocení těchto typů BI platforem je zejména způsob a rychlost zpracování dat, s možností sdílení a publikace výstupů na rozličná zařízení, variabilita možností při tvorbě v daném programu, infrastruktura řešení z hlediska přehlednosti a bezpečnosti a variabilita v přístupu importování dat do řešení. [18]

Obrázek 11: Magic Quadrant for Data Integration Tools [18]

4.2 Implementace Data Vault metodiky

Cílem této kapitole je představit metodiku modelování Data Vault na praktickém příkladu. Pro účel práce jsem zvolil volně dostupnou databázi od společnosti Oracle, konkrétně Oracle HR schéma.

4.2.1 Zvolený model

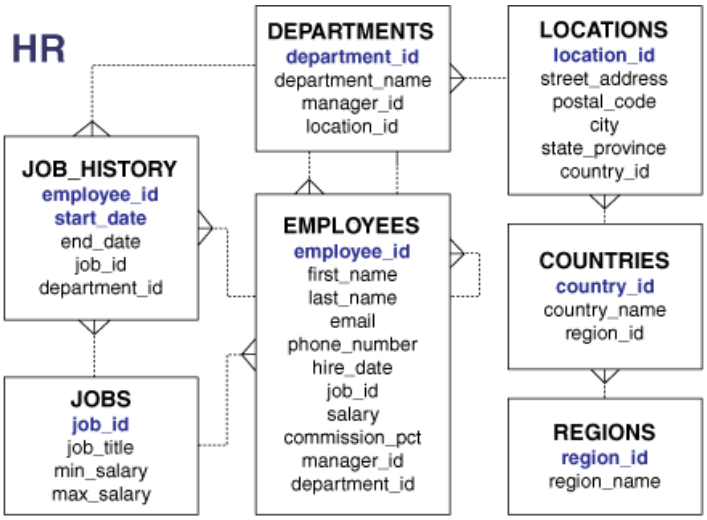
Model je znázorněn pomoci ER diagramu (Entity Relationship Diagram) neboli ERD. ER diagram znázorňuje vztahy neboli relace mezi tabulkami. Každá tabulka představuje přesně definovanou množinu dat neboli entitu. Entity obsahují atributy, které definují vlastnosti tabulky.[14]

- **Entita –** představuje libovolný logický objekt reálného světa, jako třeba osoba, věc, událost atd. Musí být rozlišitelná od ostatních entit a existovat nezávisle na nich.
- **Relace** – představuje, jakým způsobem je mezi dvěma nebo více entitami zachycen jejich vztah.
- **Kardinalita vztahů** – představuje skutečnost, kolik výskytů jedné entity může vstoupit do vztahu s kolika výskyty druhé entity. Známe tři typy vztahů
    - 1:1 – jeden záznam entity odpovídá právě jednomu záznamu druhé entity
    - 1:N – jeden záznam entity odpovídá jednomu nebo více záznamům druhé entity
    - N:M –více záznamů jedné entity odpovídá více záznamům druhé entity

Je nutno zmínit, že zvolený model je již normalizován, a to konkrétně ve třetí normalizační formě neboli 3NF. Normalizace je proces, který by měl vést k vzniku tabulek, které lze snadno a efektivně udržovat. Normalizace má několik hlavních úrovní normalizace a snahou je dosáhnout nejvyššího stupně normalizace. [15]

- **1. normalizační forma (1NF)** – relace splňuje požadavky pro první normální formu, pokud každý atribut tabulky obsahuje nedělitelné/atomické hodnoty.
- **2. normalizační forma (2NF)** – relace splňuje požadavky pro druhou normální formu, pokud splňuje podmínky 1NF a každý neklíčový atribut je plně závislí na
- primárním klíči.

- **3. normalizační forma (3NF) –** relace splňuje požadavky pro třetí normální formu, pokud splňuje-li 1NF a 2NF a všechny její neklíčové atributy jsou vzájemně nezávislé.



Obrázek 12: Oracle HR Schema [22]

4.2.2 Implementace navrhnutých změn

Metodika Data Vault je skvěle připravena pro modelování databází. Kombinuje metodiku ERD modelování a třetí normalizační formu (3NF).

Prvním krokem návrhu Data Vault modelu je identifikace základních entit a vytvoření Hub tabulek. Zvolený HR model se již splňuje podmínky 3NF a zachycuje detailní rozpad. Pro vytvoření Hub tabulek nám stačí, vytvořit jednotlivé tabulky z původního modelu s jejích business klíčem. Business klíč se plní pomoci pomocí zvolené Hash funkce a vkládáme do něj business klíč společně s ID zdrojového systému. HashKey představuje cizí klíč potřebný pro propojení s dalšíma relacemi. Dalším atributem je Surrogate klíč, který plní funkci primárního klíče (PK) tabulky a je plněný ze sekvence. Posledními atributy při vytváření Hub tabulky, jsou technické atributy časová známka spolu s ID zdrojového systému.

H_DEPARTMENT		
DEPARTMENT_SK	number(20)	PK
DEPARTMENT_HK	raw(20)	FK
DEPARTMENT_ID	number(4)	N
LOAD_DATETIME	timestamp	N
SOURCE_SYSTEM	varchar2(20)	N

Obrázek 13: Tabulka typu HUB (vlastní zpracování)

Druhým krokem je sestavění vazebních tabulek neboli tabulek typu Link, potřebných pro propojení kontextových informací mezi jednotlivýma Hub tabulkami. Link neobsahuje vlastní business klíč jako taký, ale obsahuje již za hashované business klíče entit, které představují cizí klíče (FK) tabulky. Stejně jako u Hub tabulky, Link obsahuje také Surrogate klíč, plnící funkci primárního klíče tabulky, technické atributy časová známka a ID zdrojového systému.

L_JOB_DEPARTMENT		
JOB_DEPARTMENT_SK	number(20)	PK
JOB_DEPARTMENT_HK	raw(20)	FK
DEPARTMENT_HK	raw(20)	FK
JOB_HK	raw(20)	FK
LOAD_DATEIME	timestamp	N
SOURCE_SYSTEM	varchar2(20)	N

Obrázek 14: Tabulka typu Link (vlastní zpracování)

Třetím a posledním krokem je sestavění kontextových tabulek neboli tabulek typu Satellite. Při vytváření satelitů, nastává tvořívá činnost. Příčinou je fakt, že Hub může mít vícero satelitů a rozdělení jednotlivých atributů po satelitech závisí jenom na datovém analytikovi. Přesně jako Link, satelit neobsahuje svůj vlastní business klíč, primární klíč představuje Surrogate klíč, cizím klíčem je business klíč již v zmíněné Hash formě a stejně jako oba typy tabulek obsahuje technické atributy časová známka a ID zdrojového systému.

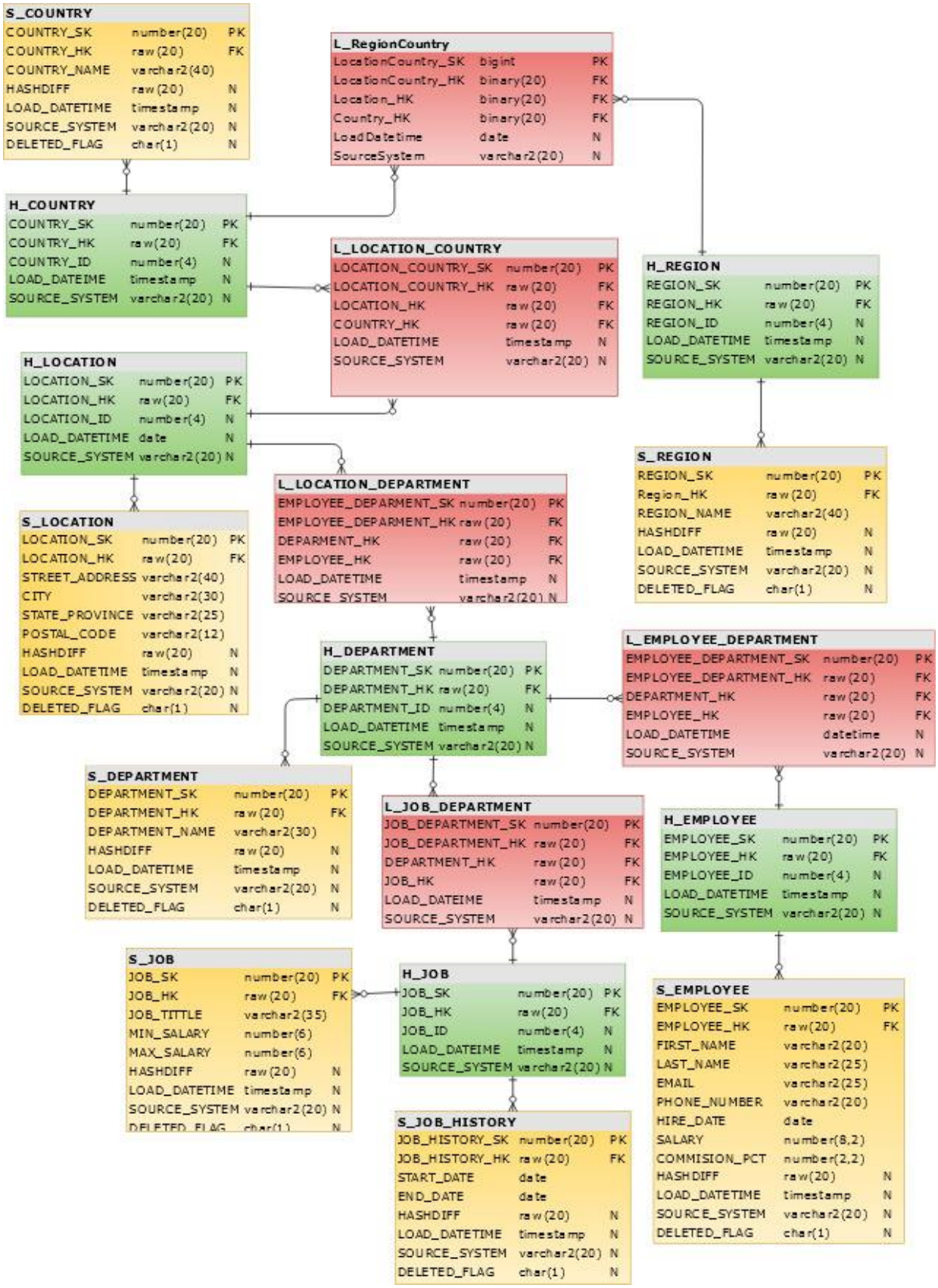
Mimo této atributy, satelit obsahuje všechny popisní informace/atributy společně pro business kontext a vztahy mezi nimi. Satelit, jako jediný konstruktor zachytává po celou historii datového skladu, proto je nutné do tabulky začlenit atributy HashDiff, do kterého se plní agregovaný obsah popisných atributů pomocí zvolené Hash funkce. Když se nám změní záznam atributu, tak se logicky také změní HashDiff a pomoci rozdílů HashDiffu jsme schopní zachytit danou změnu. Posledním důležitým atributem satelitu je Deleted Flag, který zachytává informaci, zdali se nám daný záznam nachází ve zdroji. Při

odmazání záznamu, uložíme záznamy jako nový řádek s Deleted Flag, kteří ukončí platnost záznamu k danému datu.

S_JOB		
JOB_SK	number(20)	PK
JOB_HK	raw(20)	FK
JOB_TITTLE	varchar2(35)	
MIN_SALARY	number(6)	
MAX_SALARY	number(6)	
HASHDIFF	raw(20)	N
LOAD_DATETIME	timestamp	N
SOURCE_SYSTEM	varchar2(20)	N
DELETED_FLAG	char(1)	N

Obrázek 15: Tabulka typu Satelit (vlastní zpracování)

Výsledný model po propojení jednotlivých konstruktorů je znázorněn na obrázku níže. Z původních sedm entit nám vzniklo šest tabulek typu Hub, definujících business koncept modelu. Z původního modelu tabulka JOB_HISTORY nepředstavuje samostatnou business entitu, neobsahuje svůj vlastní business klíč, a proto je konstruována jako satelit.



Obrázek 16: Data Vault HR schema (vlastní zpracování)

4.3 Integrace dat v ODI

Cílem kapitole je představit práci v integračním nástroji ODI, jednotlivé funkce a rozhraní nástroje a na praktickým příkladě předvést znalost nástroje.

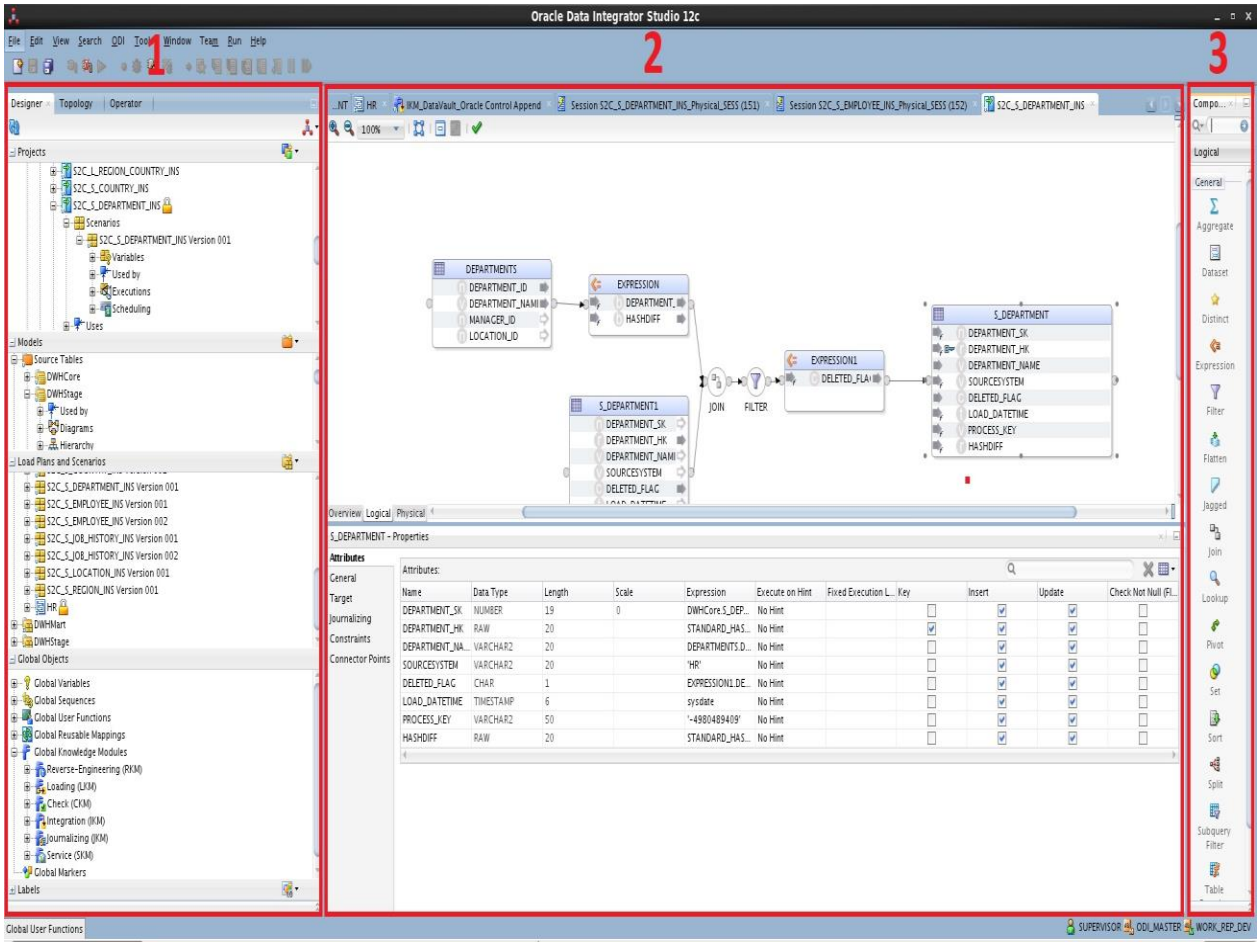
Oracle Data Integrator (ODI) disponuje širokou škálou funkcí s intuitivním grafickým rozhraním. Platforma je obsáhlá, která pokrývá všechny požadavky na integraci dat: od velkoobjemových, vysoce výkonných dávkových nahrávaní, přes integraci procesů založených na událostech až po servisně orientovanou architekturu. Mezi hlavní funkce ODI se řadí [12][13]:

- ODI není **ETL** řadí se jako nástroj **E-LT** – Rozdíl je ten, že u E-LT procesů se transformace provádí až v cílovém systému. U ETL procesů se transformace dat provádí výhradně pomoci ETL nástroje na dedikovaném servery. Tato architektura je oproštěná od zbytečných přenosu dát po síti
- **Znalostní modul** –znalostní modly určují jakým způsobem dochází k integračním procesům. Existuje několik typů znalostních modulů, které představím v další kapitole.
- **Deklarativní pravidla** – ODI vývojář může specifikovat pravidla pro integrační procesy a řízení datových toků mezi zdrojovým a cílovým systémem.

4.3.1 Práce s ODI

Práce se zabývá využitím ETL nástroje Oracle Data Integrator (ODI), proto se v této části pokusím představit způsoby využití ODI. Společnost Oracle popisuje ODI, jako obsáhlou integrační platformu, která musí splňovat vysoká kritéria potřebné pro integraci dat.

ODI disponuje s grafickým prostředím, kde lze pracovat jednoduše s různými objekty. Při tvorbě ETL mapování se dá jednoduše pomoci chyť-a-pust metody přetáhnout libovolný objekt myší do pracovní plochy a dále s nimi pracovat, kde můžeme propojovat jednotlivé objekty, jako jsou tabulky, filtry, spájení a mnoha dalších. Samozřejmě je možnost i ručního psaní.



Obrázek 17: ODI obrazovky (vlastní zpracování)

Na obr. 17 je hlavní vývojová část nástroje ODI rozdělená na 3 dílčí obrazovky.

- Na 1. obrazovce vidíme 4 hlavní prvky ODI tzv. Navigator, které představují základní části ODI – Designer, Topology, Operator a Security.
- Na 2. obrazovce vidíme pracovní plochu, pro editaci samotného mapování, kde můžeme různě editovat objekty, proměnné atd.
- 3. obrazovka obsahuje objekty, jako agregace, spájení, filtr, výrazy atd. a můžeme je pouhým přetáhnutím (drag and drop) přesunout do pracovní plochy

4.3.1.1 Designer

Navigátor designer se používá pro navrhování integritních omezení, vývoj projektu a budování transformací jako: [13][19]

- Automatický reverse-engeneering existujících aplikací nebo databází v modelu
- Grafický vývoj s údržbou rozhraní transformace a integrace
- Vizualizace datových toků v mapování
- Automatické generování dokumentace
- Přizpůsobení vygenerovaného kódu

4.3.1.2 Topology

Navigátor Topology se používá ke správě dat popisujících fyzický informační systém a logickou architekturu. Skrz topologii můžeme spravovat svůj informační systém, technologie s jejich datovými typy, datové servery propojené na dané technologie a schémata obsahující kontextové informace, jazyky, agentův a samozřejmě jednotlivé úložiště.[13][19]

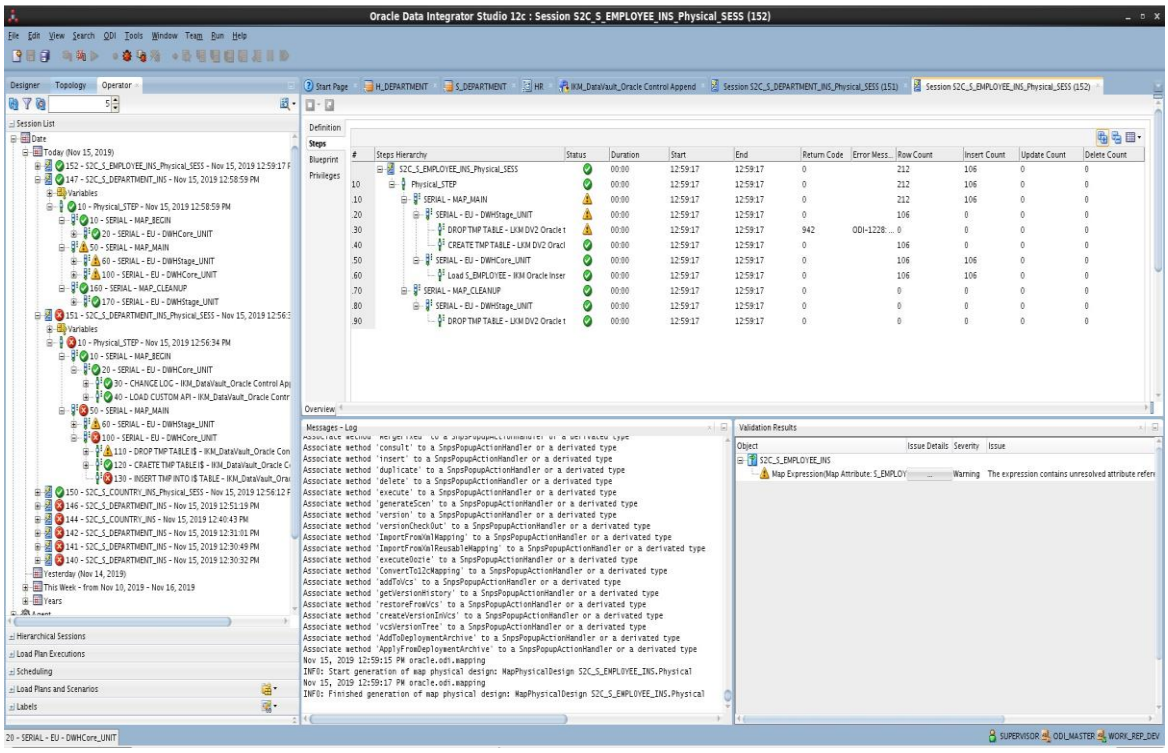
- **Fyzická architektura** – definuje různé fyzické komponenty informačního systému, sloužící pro uchovávaní a umísťování strukturovaných dat na datovém serveru. Platí, že jenom jedna technologie může být propojena s jedním datovým serverem. Datový server ukládá informace dle specifické logiky, která je uložena ve fyzické architektuře a přiřazena datovému serveru. Všechny databázové servery, JMS soubor správ (Java Message Service), různé druhy souborů jako csv a mnoha dalších, které se využívají v ODI, musí být deklarován jako datový server. Fyzická architektura obsahuje také definici fyzických agentů. Jedná se o komponenty softwaru Java, které pomáhají ODI s během Jobů.
- **Kontext –** spojuje dohromady komponenty fyzické architektury s komponenty logické architektury. Kontext může korespondovat mezi různým prostředím jako například vývojové, integrační, před produkčním a produkčním prostředím.
- **Logická architektura** – umožnuje uživateli identifikovat skupinu podobných fyzických schémat, která obsahují konstrukčně shodná datová uložiště a jsou umístěna na různých fyzických místech. Logické schéma stejně jako její protějšek fyzické schéma, jsou napojena na nějakou technologii.

Pro přiklad, Oracle logické schéma *Accounting* (účetnictví) může korespondovat se dvěma Oracle fyzickýma schémata:

- Accounting Sample používané ve vývojovém kontextu
- Accounting Corporate používané v produkčním kontextu

Tyto dvě fyzická schémata jsou shodná ve své struktuře, obě obsahují data z účetnictví, ale jsou umístněná na různém místě. Umístnění fyzických schémat je pravděpodobně na různých Oracle instancích – datových serverech. Všechny komponenty vyvinuté v ODI, jsou navrhnuté pro logickou architekturu. Například, datový model je vždy propojen s logickým schématem a datové toky jsou definované pro jejích model.

4.3.1.3 Operator



Obrázek 18: ODI běh procesů (vlastní zpracování)

Navigátor operátor je nástroj pro management produkce a monitorování běhu. operátor umožňuje řídit exekuce jednotlivých scénářů a kontrolu provedení kódu. Průběh scénářů můžeme vidět na obr. 18. Barevné ikony znázorňují průběh jednotlivých operací. Zelená barva s fajfkou představuje dokončenou operaci, světle zelený trojúhelník právě probíhající operaci, žlutý trojúhelník s vykřičníkem znamená varování a poslední červená barva s křížkem znamená chybu.

Po rozkliknutí běhu konkrétního scénáře se nám v pravé části rozbalí obrazovka, kde jsou uvedené jednotlivé kroky. ODI nám nabízí možnost se podívat na SQL kód, po rozkliknutí kroku, který nás zajímá. [13]

4.3.1.4 Security

Navigátor Bezpečnost slouží jako nástroj pro správu informací v ODI, můžeme zde vytvářet uživatele, role a profily s různým uživatelským oprávněním.[13]

4.3.1.5 Znalostní modul

V kapitole 4.3 jsem popsal znalostní moduly. ODI má v sobě několik typů znalostních modulů, každý typ odkazuje na konkrétní integrační úkol

- **Reverse-engeneering (RKM)** – načte metadata z různorodých systémů do pracovního uložiště ODI
- **CDC – Changed Data Capture (JKM)** – poskytuje čtení a identifikaci změněných dat ze zdroje a nahrává data do staging layeru.
- **Loading Knowledge Module (LKM)** – načítá a nahrává data ze zdrojů do vstupní vrstvy a poradí si s načtením a nahráváním dat mezi různýma technologiemi.
- **Integration Knowledge Module (IKM)** – integrace dat ze vstupní vrstvy do cílové oblasti. Integrační mód je nastavění používané pro integraci záznamů z datových toků do cíle. Mezi běžné integrační nastavění patří **Append** – jenom vkládaní záznamů s možností smazaní záznamů před samotným vložením. **Control Append** – stejný jako Append, ale navíc kontroluje tok dát. **Incremental Update** – stejný jako Control Append, ale navíc je možný aktualizovat záznamy datového toku. **Slowly Changing Dimension (SCD2)** – Integrace dat do tabulky pomoci 2 typu pomalu se měnící dimenze. Podporuje vkládaní, aktualizaci a mazání záznamů.
- **Controlling Knowledge Module (CKM)** – kontrola kvality integračních tocích, pokud je stage layer je umístněný na Oracle data serveru.
- **Service Knowledge Module (SKM)** – generuje kód potřebný pro vytvoření datových služeb

Nejdůležitější znalostní moduly pro vývojáře jsou LKM a IKM. Slouží jako šablona kódu, která může být použita mezi různým prostředím s různou technologií. Například LKM nám zaručí přenos mezi prostředím Hive a Oracle, tím že zkonvertuje datové typy využívané onou technologii. IKM nám umožnuje využít předem nadefinovaný způsob integrace, jako například TRUNCATE INSERT nebo MERGE a mnoha dalších. Jednotlivá

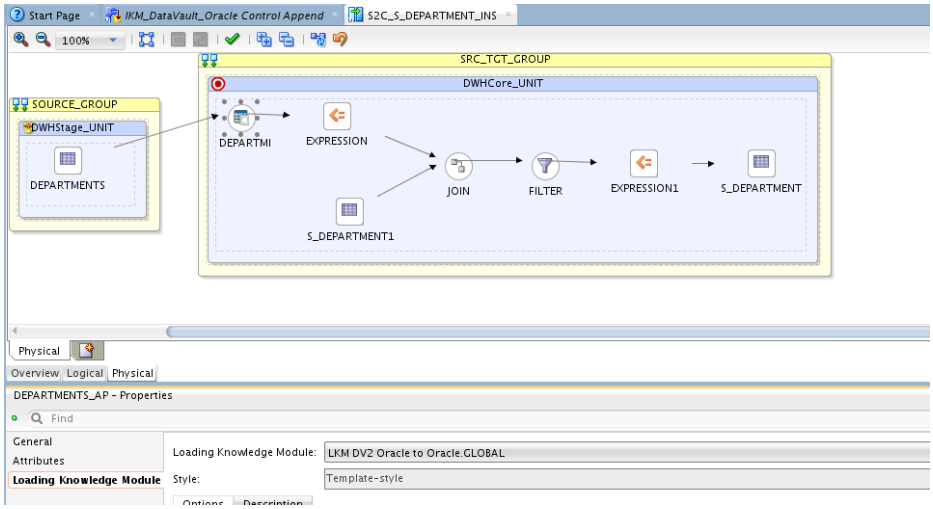
KM, jsou předdefinované od výrobce, ale samozřejmě máme možnost si definovat své vlastní. [13]

4.3.1.6 Mapování

Mapování definuje transformace vykonané z jednoho nebo vícero zdrojových sloupců pro nahrání jednoho cílového sloupce. Tyhle transformace jsou implementované ve formě SQL dotazů. [13]

4.3.1.7 Datový tok

Tok dat definuje, jak data proudí mezi zdrojovými systémy, vstupní vrstvou, pokud se liší od cíle a cílovou oblastí, kde jsou také spájeny tabulky a filtry. Datový tok také zahrnuje metody pro načítání a integrace, využívané daným rozhraním, které se následně vybírají pomoci znalostních modulů, konkrétně LKM a IKM. [13][19]



Obrázek 19: ODI změna zaznamenaných dat (vlastní zpracování)

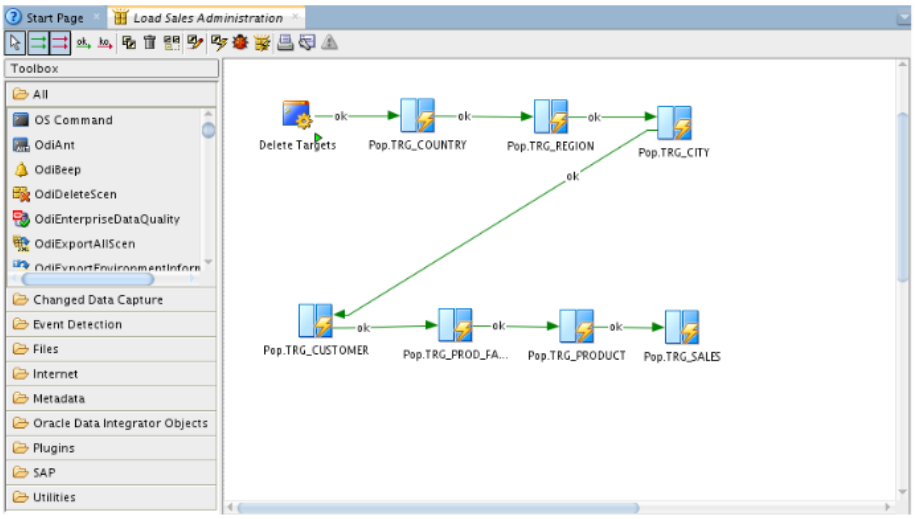
4.3.1.8 Změna zaznamenaných dat CDC

CDC umožnuje ODI sledovat změny ve zdrojových datech způsobené jinými aplikacemi. Omezení toku zdrojových dat pouze na změněná data je užitečné v mnoha kontextech, jako je synchronizace a replikace dat. [13][19]

4.3.1.9 Balíček

Balíček je největší prováděcí jednotkou v ODI. Balíček se skládá ze sekvence kroků, uspořádaných do exekučního diagramu. Každý krok může buď uspět nebo selhat při

jeho provádění. V závislosti na výsledku exekuce jednotlivých kroků (úspěch nebo neúspěch) může být krok rozvětven do jiného kroku. [13][19]



Obrázek 20: ODI balíček [13]

4.3.1.10Scénář

Scénář představuje kód, který může být generovaný z balíčku nebo procedury (mapování). Po vygenerování je scénář uložen v pracovním uložišti a kód scénáře je zamražen, proto všechny další modifikace jednotlivých komponent je neovlivní/nezmění. Scénář lze exportovat a poté importovat do jiného uložiště a použít v různých kontextech. [13][19]

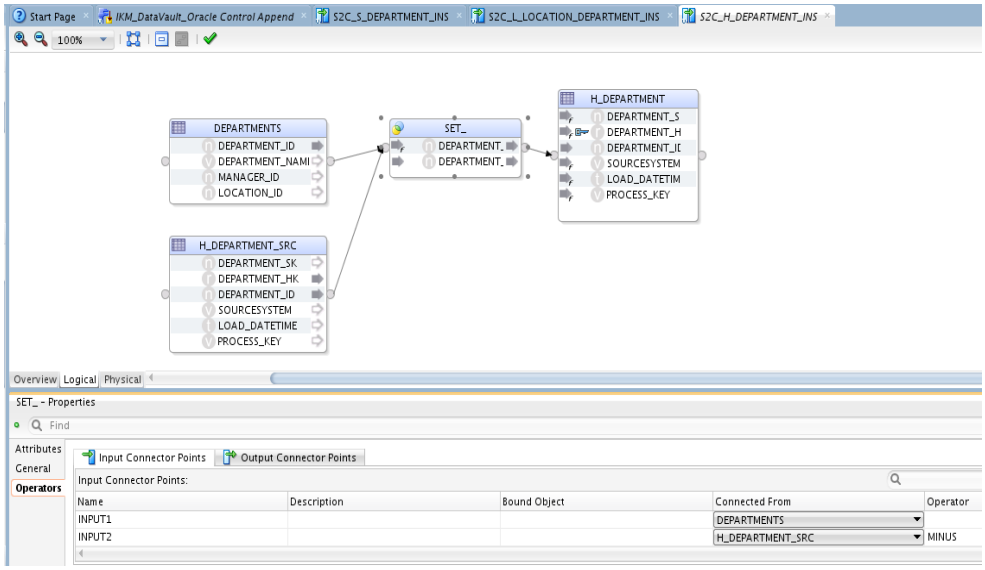
4.4 Vlastní práce s ODI

V této části bakalářské práce se zaměřím na samotné vytvoření procedur pomoci, kterých převedu původní model HR schématu do nového Data Vault HR schématu. Z důvodu rozsahu bakalářské práce uvedu příklady, jako zkonstruovat mapování plnící tabulku Hub, Link, Satellite a nebudu zde uvádět všechna mapování potřebné pro naplnění celého HR Data Vault modelu.

4.4.1 Vytvoření tabulky Hub

Mapování tohoto typu konstruktoru je jednoduché a data ze zdrojové tabulky, konkrétně business klíč převedeme 1:1 do cílové tabulky.

V logické části mapování použijeme jako zdrojové tabulky, uložené v modelech nástroje ODI, konkrétně tabulku DEPARTMENTS z HR schématu a již naplněnou H_DEPARTMENT_SRC z cílového schématu DWHCore. Zdrojové tabulky spojíme v objektě SET, kde si také vytvoříme již za hashované business klíče a nastavíme operátor na minus, tak docílím, že se mi budou vkládat jenom nově příchozí záznamy.



Obrázek 21: ODI mapování tabulky Hub (vlastní zpracování)

Atributy cílové tabulky H_DEPARTMENT se plní následovně:

- DEPARTMENT_SK – se naplní hodnotou ze sekvence.
- DEPARTMENT_HK – se plní jako agregace business klíče a ID zdrojového systému pomocí Hash funkce.
- DEPARTMENT_ID – se plní business klíčem ze zdrojové tabulky.
- SOURCE_SYSTEM – se plní konstantní hodnotou, která nám identifikuje zdrojový systém.
- LOAD_DATETIME – naplníme časovou známkou databázového serveru.

Když máme logickou část implementovanou, je potřebné nastavit fyzickou část mapování. Konkrétně se jedná o nastavění nahrávacího (LKM) a integračního (IKM) znalostního modulu.

Zvolené LKM provádí následující kroky:

- Smaže *temporary* (dočasnou) tabulku, pokud již v databázi byla založena.
- Vytvoří dočasnou tabulku a nahraje do ní záznamy.
- V posledním kroku znovu smaže dočasnou tabulku.

Důvod, proč jsem ve svém LKM zvolil na začátku a na konci promazávání dočasných tabulek je, že v případě kolizi procedury, zůstane dočasná tabulka nezmazaná a při dalším spuštěním procedury by to spadlo s chybou, že již objekt v dané databázi existuje.

Posledním krokem nastavění fyzické části mapování je jeho integrační část IKM. Zvolené IKM provádí následující kroky:

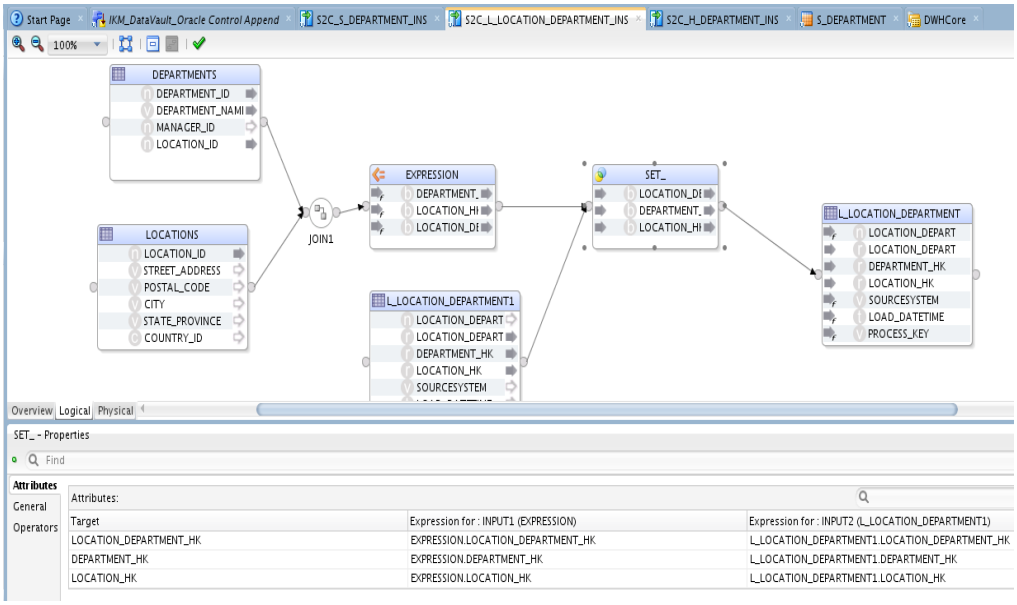
- Smaže temporary tabulku, pokud již v databázi byla založena.
- Vytvoří dočasnou tabulku.
- Nahraje záznamy do dočasné tabulky
- Nahraje nové záznamy do cílové tabulky
- V posledním kroku znovu smaže dočasnou tabulku.

Vygenerovaný SQL kód naleznete v přílohách mé bakalářské práce. Konkrétně se jedná o Příloha 1.

4.4.2 Vytvoření tabulky Link

Implementace logické a fyzické části mapováni je obdobná jako u Hub tabulky. Jediný rozdíl mezi Hubem a Linkem je ten, že do linku vkládáme za hashované Natural Business klíče a složený za hashovaný business klíč z obou entit, které slouží jako vazba mezi nimi.

V logické části mapování použijeme dvě zdrojové konkrétně se jedná o tabulky DEPARTMENTS a LOCATIONS, které se budou plnit do cílové tabulky L_JOB_DEPARTMENT, jako další zdrojovou tabulku, stejně jak u plnění tabulky Hub využijeme již naplněnou L_JOB_DEPARTMENT, kde si jednotlivé zdroje propojíme v objektu SET a nastavíme operátor na minus



Obrázek 22: ODI mapování tabulky typu Link (vlastní zpracování)

Atributy cílové tabulky L_LOCATION_DEPARTMENT se plní následovně:

- LOCATION_DEPARTMENT_SK – se naplní hodnotou ze sekvence.
- LOCATION_DEPARTMENT_HK – se plní jako agregace business klíčů z obou entit s ID zdrojového systému pomocí Hash funkce.
- DEPARTMENT_HK a LOCATION _HK – se plní jako agregace business klíče a ID zdrojového systému pomocí Hash funkce.
- SOURCE_SYSTEM – se plní konstantní hodnotou, která nám identifikuje zdrojový systém.
- LOAD_DATETIME – naplníme časovou známkou databázového serveru.

Ve fyzické části se použije stejné LKM a IKM jako u tabulky typu Hub.

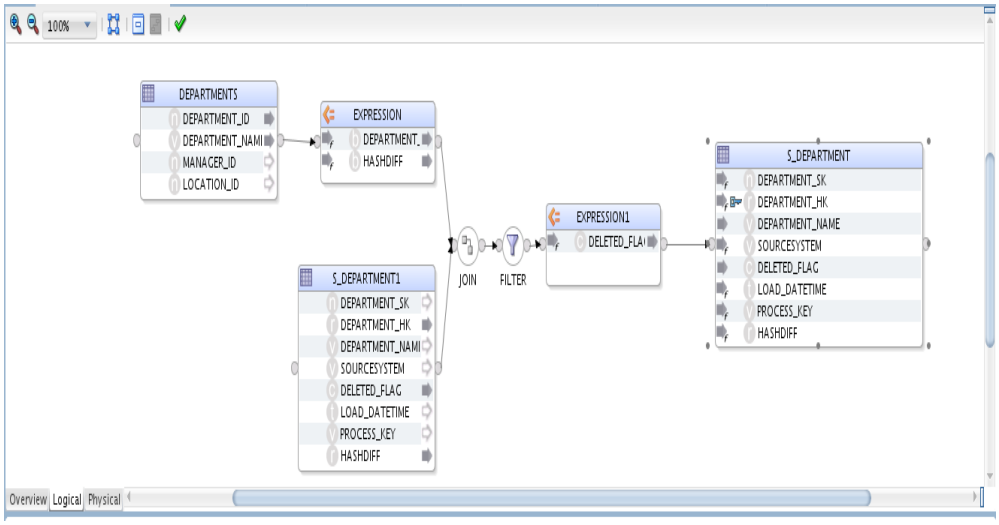
Vygenerovaný SQL kód naleznete v přílohách mé bakalářské práce. Konkrétně se jedná o Příloha 2.

4.4.3 Vytvoření tabulky Satelit

Implementace logické části oproti Hubu a Linku vyžaduje větší práci. Jako zdrojové tabulky jsem použil tabulku DEPARTMENTS a S_DEPARTMENT1. Důvodem

proč, jsem si zvolil jako druhou tabulku, do které data plním je ten, abych byl schopný zachytávat změny.

V objektu EXPRESSION vytvářím DEPARTMENT_HK stejným způsobem, jako u plnění cílové tabulky, abych byl schopný tabulky mezi sebou propojit. Stejným způsobem vytvářím HASHDIFF, jako při plnění cílové tabulky, který mi slouží pro zachytávání změn. Tabulky jsou spojeny objektem JOIN, kde jsem nastavil FULL OUTER JOIN a následně za spojením je objekt FILTER s podmínkami, které definují, jestli se jedná o nový záznam, již existující záznam nebo smazaný záznam. V posledním objektu EXPRESSION1 mám nadefinováno plnění DELETED_FLAG pomocí klauzule CASE s podmínkami, které mi definují, jestli se jedná o nový nebo změněný záznam, vložím konstantu N a když se jedná o smazaný záznam, vloží se konstanta Y. Posledním krokem implementace v logické části je na mapování popisné atributy. Technické atributy a business klíč se plní stejným způsobem jako u Hubu a Linku.



Obrázek 23: ODI mapování tabulky typu Satelit (vlastní zpracování)

Atributy cílové tabulky S_DEPARTMENT se plní následovně:

- DEPARTMENT_SK – se naplní hodnotou ze sekvence.
- DEPARTMENT_HK – se plní jako agregace business klíčů z obou entit s ID zdrojového systému pomocí Hash funkce.
- DEPARTMENT_NAME – se plní odpovídajícím kontextovým atributem.
- DELETED_FLAG – se plní pomocí klauzule case za splnění nadefinovaných podmínek.
- SOURCE_SYSTEM – se plní konstantní hodnotou, která nám identifikuje zdrojový systém.

- LOAD_DATETIME – naplníme časovou známkou databázového serveru.
- HASHDIFF – se plní pomocí Hash funkce, jako agregace všech kontextových atributů

Ve fyzické části se použije stejné LKM a IKM jako u předcházejících mapování.
Vygenerovaný SQL kód naleznete v přílohách mé bakalářské práce. Konkrétně se jedná o Příloha 3.

4.5 Návrh jednotlivých fází integrace dat

Integrace začíná obdržením zprávy s impulsem k začátku vývoje BI systému, na oddělení analytiků. Ti dále zanalyzují požadavky a formulují je pro vývojáře.

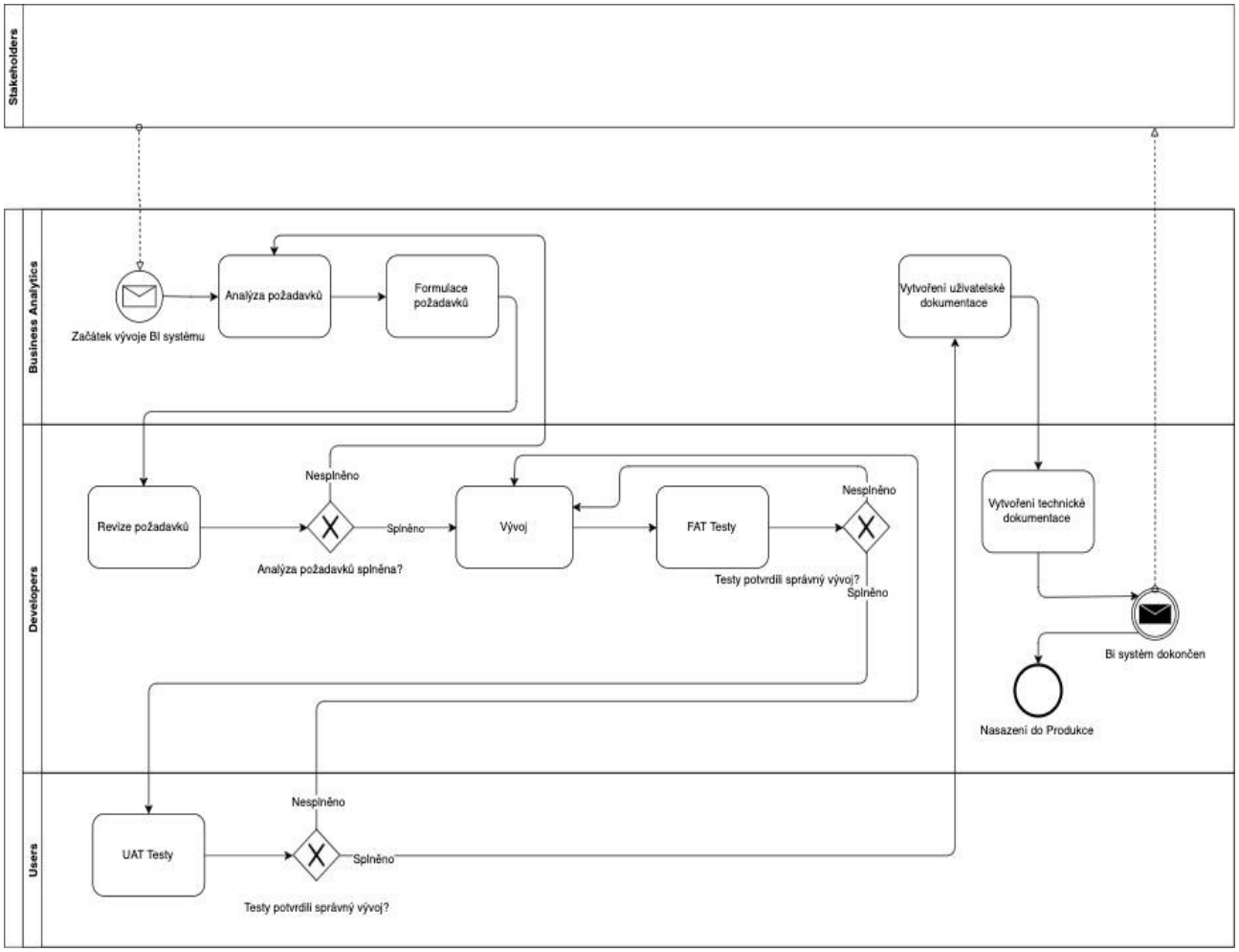
Požadavky jsou následně revidovány vývojáři, v případě zamítnutí se vrátí zpět k analýze a formulaci, cyklus se opakuje. Pokud jsou požadavky splněny, zadání je vývojáři implementováno do kódu. Po dokončení vývoje se provádí testováni tzv. FAT testy, funkční testy.

Funkční testy jsou prováděné na straně dodavatelského týmu, obvykle dalším členem dodavatelského týmu, pokud jsou funkční testy nesplněny, vrací se zpět do vývoje k vývojářům, v opačném případě se nasadí požadavek na vyšší prostředí, kde se rovnou otestuje úroveň integrace implementovaného řešení a předá se požadavek byznys uživateli k tzv. UAT testům, akceptačním testům. Akceptační testy jsou prováděné na straně zákazníka, kde provádějí testy podle předem připravených scénářů, které společné připravil zákazník s dodavatelem. Potvrzení o nesprávnosti vývoje od uživatelů opakuje cyklus vývoje. [21]

Správný vývoj je navrácen business analytikům a ti vytvoří uživatelskou dokumentaci, vývojáři vytvoří technickou dokumentaci.

Hotový a otestovaný BI systém je dokončen a zpráva je odeslána stakeholderům. Systém je nasazen do produkce.

Následný diagram je vytvoření z vlastní zkušenosti a z teoretického uvažování nad nejlepším možným postupem získání všech důležitých částí pro kvalitní BI systém. Systém by měl mít kvalitní analýzu, kód, testování kódu a testování samotného systému uživatelem. Systém vzniká vždy z potřeby, tedy z požadavků, které poskytnou v daném schématu stakeholdeři.



Obrázek 24: BPMN návrh procesu integrace (vlastní zpracování)

5 Výsledky

5.1 Zhodnocení výsledků

Výsledkem práce je jasná implementace Oracle HR Schema testovací databáze do nového modelu podle vzoru Data Vault. Data Vault ukazuje, že dekompozice na malé části zrychluje vývoj. Díky menším částem se dá vývoj nejen urychlit ale také lépe adaptovat na změny v zadání. Menší části znamenají i menší model, a tak je zde jasné zrychlení i s učením nového vzoru. Defaultně nám model umožňuje držet důležitá business data a zachovávat historií k veškerým změnám jejich parametrů. Toto umožňuje také lépe chápat business procesy.

Díky možnosti vyvíjet systém po malých částech, iteracích, je zde jasný potenciál pro agilní vývoj, tak jak je vidět v samotné práci. To ovšem znamená že každá iterace, která bude vytvářet skupinu menších částí, rapidně zvýší jejich počet. Toto může být nevýhodou dané implementace. Důležité v této části tedy je si uvědomit, na jaký projekt je Data Vault nasazován a zvážit i jiné možnosti. Z teoretického uvažování je jasné, že se nebude hodit pro všechny typy projektů, např. pro ty, které mají malý počet entit, zdrojových systémů a nemají potřebu se v dlouhém časovém horizontu rozvíjet.

Pro celou implementaci je zde kód v kapitole přílohy, konkrétně se jedná o Příloha 1, Příloha 2 a Příloha 3. Celá implementace tak poskytuje náhled nejen na teoretický proces vývoje, ale poskytuje kompletní řešení. Ukázka pro budoucí použití v podnikové praxi je jasná. Práce nenabízí komparativní analýzu vícero nástrojů, raději se zaměřuje na důkladnou studii a použití jednoho kvalitního nástroje. Jejím výsledkem je použití, nasazení a implementace takové studované technologie. Ukazuje tak na jasné výhody této technologie i její nevýhody s možností nahlédnutí do kódu, kde předlohou je firemní standard Oracle HR Schema volně dostupná databáze, proměněná s použitím Oracle Data Integrator nástrojem a metodikou Data Vault.

6 Závěr

Tato práce se zabývala problematikou Business Intelligence, využívanou zejména v podnikové praxi velkých organizací, za účelem podpory podnikových rozhodnutí, zlepšení a zkvalitnění procesů uvnitř organizace a snížení časových a finančních nákladů organizace.

Hlavním cílem této práce bylo navrhnout proces integrace dat z datového uložiště do konsolidovaného datového skladu v podnikové praxi. K dosažení tohoto cíle jsem vypracoval kritickou literární rešerši, kde se v první kapitole věnuji vymezení pojmu Business Intelligence, vývoj BI, představení BI systému a jeho řešení. Kapitola se dále zabývala popisem jednotlivých komponentů BI řešení, kde jsem představil jednotlivé datové zdroje, definoval datový sklad s jeho vrstvami, vysvětlil principy ETL. Dále jsem popsal jeden z trendů, konkrétně metodiku Agilního vývoje, zabývající se interaktivním způsobem řízení projektů, vysvětlil jsem význam dodávek produktů průběžně v malých iteracích zákazníkovi, důležitost spolupráce a zpětné vazby od zákazníka, popsal manifest definující principy agilního vývoje datových skladů a definoval jednotlivé role agilního týmu. Hlavní trend, který tvoří podstatnou část mé práce je Data Vault metodika, pro modelování datových skladů. Na rozdíl od agilního vývoje zaměřeného na řízení projektů, se Data Vault zaměřuje přímo na budování datového skladu. V kapitole popisuji principy, pravidla pro modelování datového skladu, jeho využití z pohledu integrace, historizace a výhody spojené s agilním přístupem k řízení procesů. V poslední kapitole teoretické části, jsem věnoval vysvětlení metodiky Business Process Model and Notation, která se stala standardem pro modelování procesů, popsal grafické prvky využívané v metodice.

Na začátku praktické části se věnuji dílčím cílům, které byly splněny prostřednictvím zhotovení datového modelu dle metodiky Data Vault, na základě teoretických poznatků a zkušeností získaných v praxi a implementace mapování prostřednictvím zvoleného ETL nástroje Oracle Data Integrator. Hlavním cílem práce, bylo navrhnout proces integrace dat. Tento cíl byl realizován prostřednictví studie notace BPMN, z vlastní zkušenosti a z teoretického uvažování nad nejlepším možným postupem získání všech důležitých částí pro kvalitní BI systém.

7 Seznam použitých zdrojů

[1] NOVOTNÝ, Ota, Jan POUR a David SLÁNSKÝ. *Business intelligence: jak využít bohatství ve vašich datech*. Praha: Grada, 2005. Management v informační společnosti. ISBN 80-247-1094-3.

[2] POUR, Jan, Miloš MARYŠKA a Ota NOVOTNÝ. *Business intelligence v podnikové praxi*. Praha: Professional Publishing, 2012. ISBN ISBN978-80-7431-065-2.

[3] GÁLA, Libor, Jan POUR a Prokop TOMAN. *Podniková informatika: počítačové aplikace v podnikové a mezipodnikové praxi*, *technologie informačních systémů, řízení a rozvoj podnikové informatiky*. Praha: Grada, 2006. Management v informační společnosti. ISBN 80-247-1278-4.

[4] KIMBALL, Ralph a Joe CASETRA. *Data Warehouse ETL Toolkit: Practical Techniques for Extracting, Cleaning, Conforming and Delivering Data*. John Wiley, 2011. ISBN 978-0-7645-6757-5.

[5] ŠTOCHOVÁ, Zuzana a Eduard KUNCE. *Agilní metody řízení projektu*. Brno: Computer Press, 2014. ISBN 978-80-251-4196-6.

[6] INMON, W.H. a Daniel LINSTEDT. *DATA ARCHITECTURE: A PRIMER FOR THE DATA SCIENTIST Big Data, Data Warehouse and Data Vault*. 225 Wyman Street, Waltham, MA: Morgan Kaufmann, 2015. ISBN 978-0-12-802044-9.

[7] LINSTEDT, Daniel. Data Vault Basics. *Www.danlinstedt.com* [online]. 2015 [cit. 2019-11-15]. Dostupné z: https://danlinstedt.com/solutions-2/data-vault-basics/

[8] HULTGREN, Hans. *Modeling the agile data warehouse with data vault*. New Hamilton, 2012. ISBN 978-0615723082.

[9] VITOUŠEK, Michal. Data Vault 2.0: Alternativní datový model je budoucnost v realizaci datových skladů, tvrdí odborníci ze Sophia Solutions. *Www.systemonline.cz* [online]. 2018 [cit. 2019-11-14]. Dostupné z: https://www.systemonline.cz/clanky/data-vault-2.0-alternativni-datovy-model-datoveho-skladu.htm

[10] SCHNIDER, Dani. Data Vault Partitioning Strategies: White Paper. *Www.trivadis.com* [online]. [cit. 2019-11-14]. Dostupné z: https://danischnider.files.wordpress.com/2018/02/wp_data_vault_partitioning_strategies_v2.pdf

[11] GRAZIANO, Kent. Agile Modeling: Not an Option Anymore. *Agile Modeling: Not an Option Anymore* [online]. 2019 [cit. 2019-11-15]. Dostupné z: https://www.vertabelo.com/blog/data-vault-series-agile-modeling-not-an-option-anymore/

[12] SOLOMKA, Petr a Hana KLÍČOVÁ. *Informační systémy v podnikové praxi*. 2. aktualizované a rozšířené vydání. Brno: Computer Press, 2010. ISBN 978-80-251-2878-7.

[13] MIQUEL, Laura Hofman. Oracle® Fusion Middleware Developer's Guide for Oracle Data Integrator: 11g Release 1 (11.1.1). *Www.docs.oracle.com* [online]. ORACLE, 2010 [cit. 2019-11-15]. Dostupné z: https://docs.oracle.com/cd/E17904_01/integrate.1111/e12643.pdf

[14] Entity Relationship Diagram. *Www.smartdraw.com* [online]. SmartDraw [cit. 2019-11-24]. Dostupné z: https://www.smartdraw.com/entity-relationship-diagram/

[15] Modelování databází. *Www.root.cz* [online]. 2002 [cit. 2019-11-21]. Dostupné z: https://www.root.cz/clanky/modelovani-databazi/

[16] Business Process Model And Notation. *OMG Object Management Group* [online]. Needham, MA, 2011 [cit. 2019-11-15]. Dostupné z: https://www.omg.org/spec/BPMN/2.0/

[17] Magic Quadrant for Data Management Solutions for Analytics. *Www.gartner.com* [online]. Gartner, 2018 [cit. 2019-11-20]. Dostupné z: https://www.gartner.com/doc/reprints?id=1-3U1LC65&ct=170222&st=sb

[18] Magic Quadrant for Data Integration Tools and the 2019. *Www.informatica.com* [online]. Gartner, 2019 [cit. 2019-11-20]. Dostupné z: https://www.informatica.com/data-integration-magic-quadrant.html#fbid=L-3epJQ8XIe

[20] Business Process Model And Notation. *OMG Object Management Group* [online]. Needham, MA, 2011 [cit. 2019-11-15]. Dostupné z: https://www.omg.org/spec/BPMN/2.0/

[21] HLAVA, Tomáš. Fáze a úrovně provádění testů. *Testovanisoftwaru.cz* [online]. 2011 [cit. 2019-11-26]. Dostupné z: http://testovanisoftwaru.cz/tag/fat/#fat

[22] Oracle Database Online Documentation 11g Release 2 (11.2). *Www.oracle.com* [online]. Oracle [cit. 2019-11-29]. Dostupné z: https://docs.oracle.com/cd/E11882_01/server.112/e40540/tablecls.htm#CNCPT010

8 Přílohy

Příloha 1: Mapping H_DEPARTMENT 55
Příloha 2: Mapping L_LOCATION_DEPARTMENT 57
Příloha 3: Mapping S_DEPARTMENT 59

```
--DWHStage Unit, nacteni zdojovych dat na urovni LKM
   --DROP C$ TMP TABLE
   drop table ETL_OWNER.C$_0ARDOV003SKKLP1P86AN1M8MCOL purge

   --CREATE C$ TMP TABLE
   create table ETL_OWNER.C$_0ARDOV003SKKLP1P86AN1M8MCOL
    as
   select * from (
   select /*+   */
     DEPARTMENTS.DEPARTMENT_ID  DEPARTMENT_ID
   from  DWHSTAGE.DEPARTMENTS DEPARTMENTS
   where (1=1)
   )

--DWHCore Unit, transformace dat na urovnni IKM
   --DROP I$ TMP TABLE
   /*dropne docasnou tabulku*/
   drop table ETL_OWNER.I$_HA2DA6H7L5EP1N38C6JHMU7TOGN purge

    --CREATE I$ TMP TABLE
    /*vytvori docasnou tanulku*/
   create table ETL_OWNER.I$_HA2DA6H7L5EP1N38C6JHMU7TOGN
   (
     DEPARTMENT_SK   NUMBER(19,0) NULL,
     DEPARTMENT_HK   RAW(20) NULL,
     DEPARTMENT_ID   NUMBER(4,0) NULL,
     SOURCESYSTEM    VARCHAR2(20) NULL,
     PROCESS_KEY     VARCHAR2(50) NULL
   )
   nologging

   --INSERT INTO I$ TABLE
      /*vlozi data do tabulky*/
   insert /*+ append */ into          ETL_OWNER.I$_HA2DA6H7L5EP1N38C6JHMU7TOGN
   (
         DEPARTMENT_SK,
     DEPARTMENT_HK,
     DEPARTMENT_ID,
     SOURCESYSTEM,
     PROCESS_KEY
   )
   select  /*+   */
     DWHCore.H_DEPARTMENT_S.NEXTVAL DEPARTMENT_SK,
     STANDARD_HASH(CONCAT(SET_.DEPARTMENT_ID,'HR'),'SHA1') DEPARTMENT_HK,
     SET_.DEPARTMENT_ID DEPARTMENT_ID,
     'HR' SOURCESYSTEM,
     '-840894094' PROCESS_KEY
   from  ((
   SELECT
     STANDARD_HASH(CONCAT(DEPARTMENTS_A.DEPARTMENT_ID,'HR'),'SHA1')                AS
DEPARTMENT_HK ,
```

```
    DEPARTMENTS_A.DEPARTMENT_ID AS DEPARTMENT_ID
  FROM
    ETL_OWNER.C$_0ARDOV003SKKLP1P86AN1M8MCOL DEPARTMENTS_A
  MINUS
  SELECT
    H_DEPARTMENT_SRC.DEPARTMENT_HK AS DEPARTMENT_HK ,
    H_DEPARTMENT_SRC.DEPARTMENT_ID AS DEPARTMENT_ID
  FROM
    DWHCORE.H_DEPARTMENT H_DEPARTMENT_SRC
    ) SET_)
  where (1=1)

  --INSERT INTO TARGET TABLE
    /*vlozi data do target tabulky*/
  insert /*+ APPEND */ into         DWHCORE.H_DEPARTMENT
  (
        DEPARTMENT_SK,
    DEPARTMENT_HK,
    DEPARTMENT_ID,
    SOURCESYSTEM,
    PROCESS_KEY
         )
  select
        DEPARTMENT_SK,
    DEPARTMENT_HK,
    DEPARTMENT_ID,
    SOURCESYSTEM,
    PROCESS_KEY
  from  ETL_OWNER.I$_HA2DA6H7L5EP1N38C6JHMU7TOGN
--DWHCore Unit Cleaning
/*smaze docasne tabulky na urovni IKM*/
declare
v_meta_tab_cnt integer;
v_tab_cnt integer;
begin
select count(1) into v_meta_tab_cnt from all_tables
where owner = 'DWHCORE‘ and table_name = 'ODI_TMP_PURGE_DISABLE' ;
if  v_meta_tab_cnt = 0 then
   execute immediate 'drop table ETL_OWNER.I$_HACAPSGROK28PFI5H6RVTHUC684 purge';
 else
   select count(1) into v_tab_cnt from dwhcore.ODI_TMP_PURGE_DISABLE
   where OTPD_SCENARIO_NAME = 'S2C_H_DEPARTMENT_INS';
   if v_tab_cnt = 0 then
     execute immediate 'drop table ETL_OWNER.I$_HACAPSGROK28PFI5H6RVTHUC684 purge';
   end if;
end if;
end;
```

Příloha 1: Mapping H_DEPARTMENT

```
--DWHStage Unit, nacteni zdojovych dat na urovni LKM

   --DROP C$ TMP TABLE
   drop table ETL_OWNER.C$_0AL5S36I9EKPMR057660K74HCI8 purge

   --CREATE C$ TMP TABLE
      create table ETL_OWNER.C$_0AL5S36I9EKPMR057660K74HCI8
    as
   select * from (
   select /*+   */
     DEPARTMENTS.DEPARTMENT_ID  DEPARTMENT_ID,
     LOCATIONS.LOCATION_IDLOCATION_ID
   from  DWHSTAGE.DEPARTMENTS DEPARTMENTS
  INNER JOIN  DWHSTAGE.LOCATIONS LOCATIONS
       ON  DEPARTMENTS.LOCATION_ID=LOCATIONS.LOCATION_ID
   where(1=1)
   )
--DWHCore Unit, transformace dat na urovnni IKM

   --DROP I$ TMP TABLE
   /*dropne docasnou tabulku*/
  drop table ETL_OWNER.I$_LA0EAI8NCL0KR6J9O7VF9N489SB purge

    --CREATE I$ TMP TABLE
    /*vytvori docasnou tanulku*/
      create table ETL_OWNER.I$_LA0EAI8NCL0KR6J9O7VF9N489SB
   (
     LOCATION_DEPARTMENT_SK        NUMBER(19,0) NULL,
     LOCATION_DEPARTMENT_HK        RAW(20) NULL,
     DEPARTMENT_HK   RAW(20) NULL,
     LOCATION_HK         RAW(20) NULL,
     SOURCESYSTEM      VARCHAR2(20) NULL,
     LOAD_DATETIME    TIMESTAMP(6) NULL,
     PROCESS_KEY         VARCHAR2(50) NULL
   ) nologging

   --INSERT INTO I$ TABLE
      /*vlozi data do tabulky*/
      insert /*+ append */ into        ETL_OWNER.I$_LA0EAI8NCL0KR6J9O7VF9N489SB
   (
         LOCATION_DEPARTMENT_SK,
     LOCATION_DEPARTMENT_HK,
     DEPARTMENT_HK,
     LOCATION_HK,
     SOURCESYSTEM,
     LOAD_DATETIME,
     PROCESS_KEY
   )
   select  /*+   */
     dwhcore.L_LOCATION_DEPARTMENT_S.NEXTVAL LOCATION_DEPARTMENT_SK,
     SET_.LOCATION_DEPARTMENT_HK LOCATION_DEPARTMENT_HK,
     SET_.DEPARTMENT_HK DEPARTMENT_HK,
     SET_.LOCATION_HK LOCATION_HK,
     'HR' SOURCESYSTEM,
     sysdate LOAD_DATETIME,
     '-98435' PROCESS_KEY
   from  ((
   SELECT
  STANDARD_HASH(CONCAT(JOIN1_A.LOCATION_ID,JOIN1_A.DEPARTMENT_ID),'SHA1')) AS
LOCATION_DEPARTMENT_HK ,
```

```
   (STANDARD_HASH(CONCAT(JOIN1_A.DEPARTMENT_ID,'HR'),'SHA1')) AS DEPARTMENT_HK ,
    (STANDARD_HASH(CONCAT(JOIN1_A.LOCATION_ID,'HR'),'SHA1')) AS LOCATION_HK
   FROM
    ETL_OWNER.C$_0AL5S36I9EKPMR057660K74HCI8 JOIN1_A
   MINUS
   SELECT
    L_LOCATION_DEPARTMENT1.LOCATION_DEPARTMENT_HK AS
LOCATION_DEPARTMENT_HK ,
    L_LOCATION_DEPARTMENT1.DEPARTMENT_HK AS DEPARTMENT_HK ,
    L_LOCATION_DEPARTMENT1.LOCATION_HK AS LOCATION_HK
   FROM
    DWHCORE.L_LOCATION_DEPARTMENT L_LOCATION_DEPARTMENT1
    ) SET_)
   where (1=1)


    /*vlozi data do target tabulky*/
    insert /*+ APPEND */ into      DWHCORE.L_LOCATION_DEPARTMENT
    (
          LOCATION_DEPARTMENT_SK,
     LOCATION_DEPARTMENT_HK,
     DEPARTMENT_HK,
     LOCATION_HK,
     SOURCESYSTEM,
     LOAD_DATETIME,
     PROCESS_KEY
          )
    select
     LOCATION_DEPARTMENT_SK,
     LOCATION_DEPARTMENT_HK,
     DEPARTMENT_HK,
     LOCATION_HK,
     SOURCESYSTEM,
     LOAD_DATETIME,
     PROCESS_KEY
    from ETL_OWNER.I$_LA0EAI8NCL0KR6J9O7VF9N489SB

--DWHCore Unit Cleaning
   /*smaze temporarz objekty na urovni IKM*/
    declare
 v_meta_tab_cnt integer;
 v_tab_cnt integer;
 begin
 select count(1) into v_meta_tab_cnt from all_tables
 where owner = 'DWHCORE‘ and table_name = 'ODI_TMP_PURGE_DISABLE' ;
 if  v_meta_tab_cnt = 0 then
    execute immediate 'drop table ETL_OWNER.I$_LA0EAI8NCL0KR6J9O7VF9N489SB purge';
  else
    select count(1) into v_tab_cnt from dwhcore.ODI_TMP_PURGE_DISABLE
    where OTPD_SCENARIO_NAME = 'S2C_L_LOCATION_DEPARTMENT_INS';
    if v_tab_cnt = 0 then
     execute immediate 'drop table ETL_OWNER.I$_LA0EAI8NCL0KR6J9O7VF9N489SB purge';
    end if;
 end if;
 end;
```

Příloha 2: Mapping L_LOCATION_DEPARTMENT

```
--DWHStage Unit, nacteni zdojovych dat na urovni LKM
   --DROP C$ TMP TABLE
  drop table ETL_OWNER.C$_0A5INV9JML3ERD3LC7NQDL6KSEL purge

   --CREATE C$ TMP TABLE
        create table ETL_OWNER.C$_0A5INV9JML3ERD3LC7NQDL6KSEL
    as
   select * from (
   select /*+   */
     DEPARTMENTS.DEPARTMENT_ID  DEPARTMENT_ID,
     DEPARTMENTS.DEPARTMENT_NAME       DEPARTMENT_NAME
   from  DWHSTAGE.DEPARTMENTS DEPARTMENTS
   where (1=1)
   )

--DWHCore Unit, transformace dat na urovnni IKM
   --DROP I$ TMP TABLE
   /*dropne docasnou tabulku*/
  drop table ETL_OWNER.I$_LA0EAI8NCL0KR6J9O7VF9N489SB purge

    --CREATE I$ TMP TABLE
    /*vytvori temporary tabulku*/
   create table ETL_OWNER.I$_SAKJ2UHCEM6DUS07P7PPAO8DGJL
   (
     DEPARTMENT_SK    NUMBER(19,0) NULL,
     DEPARTMENT_HK    RAW(20) NULL,
     DEPARTMENT_NAME          VARCHAR2(20) NULL,
     SOURCESYSTEM      VARCHAR2(20) NULL,
     PROCESS_KEY         VARCHAR2(50) NULL,
     HASHDIFF    RAW(20) NULL
   )
   nologging

   --INSERT INTO I$ TABLE
       /*vlozi data do tabulky*/
           insert /*+ append */ into   ETL_OWNER.I$_SAKJ2UHCEM6DUS07P7PPAO8DGJL
      (
            DEPARTMENT_SK,
        DEPARTMENT_HK,
        DEPARTMENT_NAME,
        SOURCESYSTEM,
        PROCESS_KEY,
        HASHDIFF
      )
      select  /*+   */
        DWHCore.S_DEPARTMENT_S.NEXTVAL DEPARTMENT_SK,
        STANDARD_HASH(CONCAT(DEPARTMENTS_A.DEPARTMENT_ID,'HR'),'SHA1')
DEPARTMENT_HK,
        DEPARTMENTS_A.DEPARTMENT_NAME DEPARTMENT_NAME,
        'HR' SOURCESYSTEM,
        '-4980489409' PROCESS_KEY,
        STANDARD_HASH(DEPARTMENTS_A.DEPARTMENT_NAME,'SHA1') HASHDIFF
      from          (ETL_OWNER.C$_0A5INV9JML3ERD3LC7NQDL6KSEL DEPARTMENTS_A
 FULL OUTER JOIN  DWHCORE.S_DEPARTMENT S_DEPARTMENT1
          ON
(STANDARD_HASH(CONCAT(DEPARTMENTS_A.DEPARTMENT_ID,'HR'),'SHA1'))=S_DEPARTMENT
1.DEPARTMENT_HK
       )
      where         (1=1)
       and (S_DEPARTMENT1.DEPARTMENT_HK is null
```

```
      or (STANDARD_HASH(CONCAT(DEPARTMENTS_A.DEPARTMENT_ID,'HR'),'SHA1')) is null
      or
(((STANDARD_HASH(DEPARTMENTS_A.DEPARTMENT_NAME,'SHA1'))<>S_DEPARTMENT1.HASH
DIFF or S_DEPARTMENT1.DELETED_FLAG = 'Y')
            and S_DEPARTMENT1.DEPARTMENT_HK is not null
            and  (STANDARD_HASH(CONCAT(DEPARTMENTS_A.DEPARTMENT_ID,'HR'),'SHA1')) is
not null))

    //*vlozi data do target tabulky*/
    insert /*+ APPEND */ into      DWHCORE.S_DEPARTMENT
    (
        DEPARTMENT_SK,
      DEPARTMENT_HK,
      DEPARTMENT_NAME,
      SOURCESYSTEM,
      PROCESS_KEY,
      HASHDIFF
         )
    select
        DEPARTMENT_SK,
      DEPARTMENT_HK,
      DEPARTMENT_NAME,
      SOURCESYSTEM,
      PROCESS_KEY,
      HASHDIFF
    from ETL_OWNER.I$_SAKJ2UHCEM6DUS07P7PPAO8DGJL


--DWHCOre Unit Cleaning

    declare
   v_meta_tab_cnt integer;
   v_tab_cnt integer;
   begin
   select count(1) into v_meta_tab_cnt from all_tables
   where owner = 'DWHCORE' and table_name = 'ODI_TMP_PURGE_DISABLE' ;
   if  v_meta_tab_cnt = 0 then
      execute immediate 'drop table ETL_OWNER.C$_0A5INV9JML3ERD3LC7NQDL6KSEL purge';
     else
      execute immediate 'select count(1) from dwhcore.ODI_TMP_PURGE_DISABLE;
      where OTPD_SCENARIO_NAME = "S2C_S_DEPARTMENT_INS"' into v_tab_cnt;
      if v_tab_cnt = 0 then
        execute immediate 'drop table ETL_OWNER.C$_0A5INV9JML3ERD3LC7NQDL6KSEL purge';
      end if;
   end if;
   end;
```

Příloha 3: Mapping S_DEPARTMENT