Univerzita Hradec Králové

Fakulta informatiky a managementu

Katedra informatiky a kvantitativních metod

Provoz datového skladu

Bakalářská práce

Autor: Zdeněk Šorf
Studijní obor: Aplikovaná informatika

Vedoucí práce: Ing. Barbora Tesařová, Ph.D.

Odborný konzultant: Ing. Jakub Snášel

Hradec Králové 2019

Prohlášení:

Prohlašuji, že jsem bakalářskou práci zpracoval samostatně a s použitím uvedené literatury.

V Hradci Králové dne 13.11.2019 Zdeněk Šorf

Poděkování:

Děkuji vedoucí bakalářské práce Ing. Barboře Tesařové, Ph.D. za odbornou pomoc při psaní bakalářské práce. Dále děkuji Ing. Jakubovi Snášelovi za praktické rady a spolupracující společnosti za umožnění vypracování praktické části.

Abstrakt

Tato bakalářská práce se zabývá datovými sklady a jejich komponenty. V první části je vysvětlena problematika týkající se databázových systému. Dále jsou popsány datové sklady, základní charakteristika, jednotlivé architektury datových skladů a na závěr vysvětlen proces ETL. Praktická část je zaměřena na analýzu analytického databázového modelu a návrh ETL procesu spolupracující společnosti.

Klíčová slova

Datový sklad, ETL, datový trh, databáze, OLTP, OLAP

Abstract

This bachelor thesis deals with data warehouses and their components. In the first part issues of database systems are explained. Next, data warehouses, basic characteristics, individual architectures of data warehouse are described, and last, ETL process is explained. The practical part focuses on the analysis of the database model and the design for the ETL process for the cooperative company.

Keywords

Data warehouse, ETL, data mart, database, OLTP, OLAP

Obsah

1 Úvod

V dnešní době nalezneme informační systémy už v každé firmě. S rozvojem informačních technologií zároveň přibývá velký objem podnikových dat. Kam ale data ukládat a jak poznat, která jsou pro nás důležitá? Cenné informace získané z dat jsou pro firmu nesmírně důležité, jak pro správné rozhodnutí budoucího růstu nebo zvýšení celkových zisků firmy a konkurenceschopnosti. Kvůli správnosti určení, která data jsou pro nás cenná jsou tu systémy pro podporu rozhodování. Mezi systémy patří i Business Intelligence a datový sklad, který je toho součástí. Právě datovým skladem se bude tato práce zabývat.

Práce je rozdělená celkem na tři části. V první části jsou vysvětleny modely databázových systémů a rozebrány různé přístupy k řešení datového skladu. Dále jsou vysvětleny přístupy ke zpracování dat.

Ve druhé části je popsán analytický datový model datového skladu. Cílem této práce je model zanalyzovat a navrhnout vlastní řešení. Ve třetí části jsou popsány návrhy řešení ETL procesů. Procesy budou vytvořeny v prostředí Microsoft SQL Server Integration Services. Návrh procesů a implementace bude detailně popsána.

2 OLTP a OLAP systémy

Databázové systémy můžeme rozdělit na transakční (OLTP) a analytické (OLAP). Obecně lze předpokládat, že transakční systémy poskytují zdrojová data datovým skladům, zatímco analytické systémy je pomáhají analyzovat.

2.1 OLTP

OLTP (anglicky On-line Transaction Processing) systémy uchovávají záznamy o uskutečněných transakcích a jsou realizovány obvykle pomocí relační databáze. Data se získávají z provozních systémů v reálném čase a následně se ukládají do relační databáze. Schéma datového modelu relační databáze je většinou ve třetí normální formě. OLTP se vyznačuje velkým množstvím krátkých on-line transakcí jako například INSERT, UPDATE nebo DELETE. Hlavní důraz OLTP systémů je kladen na velmi rychlé zpracování dotazů a zachování integrity dat v multi-přístupovém prostředí. Účinnost OLTP systémů se měří v počtu transakcí za sekundu. V OLTP databázi jsou data detailní a aktuální.

2.2 OLAP

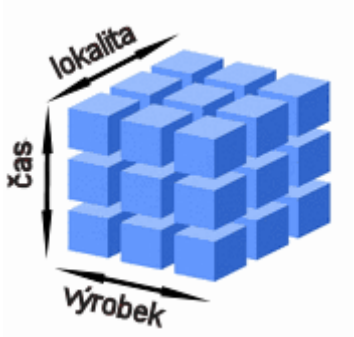
OLAP (anglicky On-line Analytical Processing) systémy jsou charakterizovány malým počtem transakcí. OLAP systémy jsou založeny na multi-dimenzionálních databázích a jsou primárně určeny pro podporu dotazování. Datový model OLAP databáze je dimenzionální, většinou se používá termín „Schéma hvězdy“ nebo „Schéma sněhové vločky“. OLAP databáze představují jednu nebo několik souvisejících OLAP kostek, které zahrnují předzpracované agregace dat podle definovaných hierarchických struktur dimenzí a jejich kombinací (1). Technologie OLAP se realizuje v několika variantách:

1. **MOLAP** (Multidimensional OLAP) je charakterizována speciálním uložením dat v multidimenzionálních – binárních OLAP kostkách.
2. **ROLAP** (Relational OLAP) je charakterizována uložením dat v relační databázi.
3. **HOLAP** (Hybrid OLAP) je kombinací MOLAP a ROLAP, detailní data jsou uložena v relační databázi a agregované hodnoty jsou uloženy v binárních OLAP kostkách
4. **DOLAP** (Desktop OLAP) umožňuje připojit se k centrálnímu úložišti OLAP dat a stáhnout si potřebnou podmnožinu kostky na lokální počítač. Veškeré

analytické operace jsou pak prováděny nad touto lokální kostkou, takže uživatel nemusí být připojen k serveru. Toto je výhodné zejména pro mobilní aplikace (1).

2.2.1 Multidimenzionální databáze

Multidimenzionální databáze nám slouží pro vytvoření více pohledů na data uložená v databázi. Na rozdíl od relační databáze, kde se data ukládají do normalizovaných tabulek, se používají převážně nenormalizované tabulky. Nenormalizované tabulky jsou tabulky, které nesplňují pravidla alespoň třetí normální formy. Data jsou uložena v multidimenzionální struktuře, v takzvané krychli nebo kostce. Multidimenzionální krychle je složena z dimenzí. Počet dimenzí může být větší než tři. Například MS SQL Server podporuje použití až 64 dimenzí.



Obrázek 1: Multidimenzionální krychle. Zdroj: (2).

Zde máme ukázku multidimenzionální krychle se třemi dimenzemi: lokalita, čas a výrobek.

Základní operace s OLAP kostkou (3)

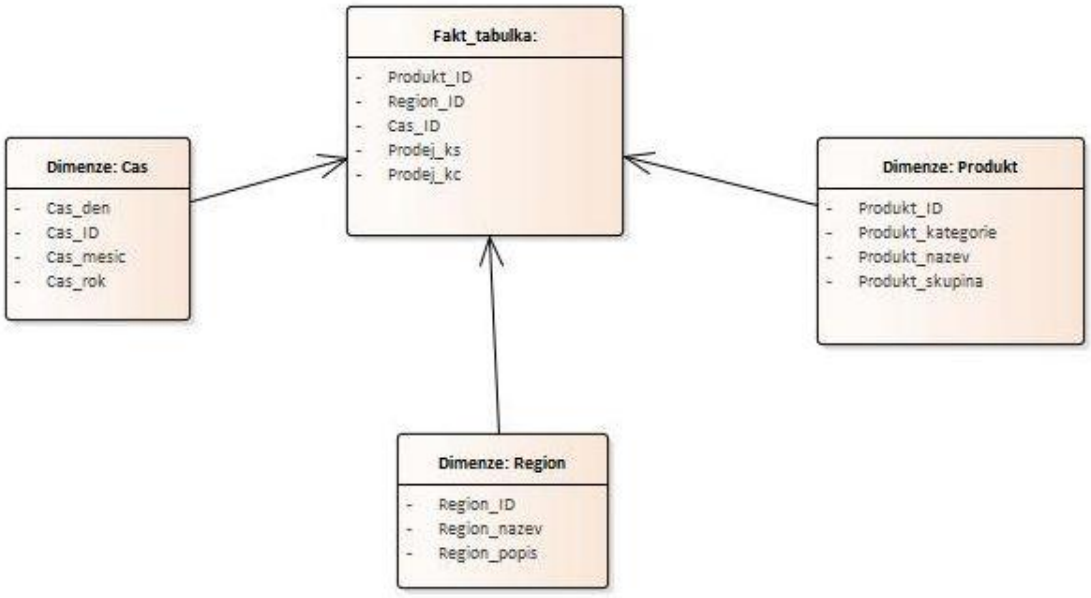
- **drill-down –** přístup k hierarchicky detailnějším datům, položkám;
- **roll-up –** přístup k hierarchicky obecnějším datům, položkám;
- **slicing –** umožňuje provádět řezy kostkou, to znamená vymezení se v jedné dimenzi na jeden prvek;
- **dicing –** slouží stejně jako operace slice, navíc jde použít pro více dimenzí;
- **pivoting –** tato operace umožňuje otáčet kostkou k získání odlišeného pohledu na stejná data.

2.2.1.1 Datový model

Datové modely relačních databází obsahují většinou mnoho tabulek a jejich vztahy. Existuje mnoho způsobů, jak se dotazovat do databáze. Odlišné dotazy mohou poskytnout stejné výsledky, tudíž pro některé uživatele to může být nepřehledné. Pro zjednodušení ERD diagramu vznikly dvě základní schémata dimenzionálního modelu: schéma hvězdy a schéma sněhové vločky. Informace v této kapitole jsou čerpány ze zdroje (1).

Schéma hvězdy

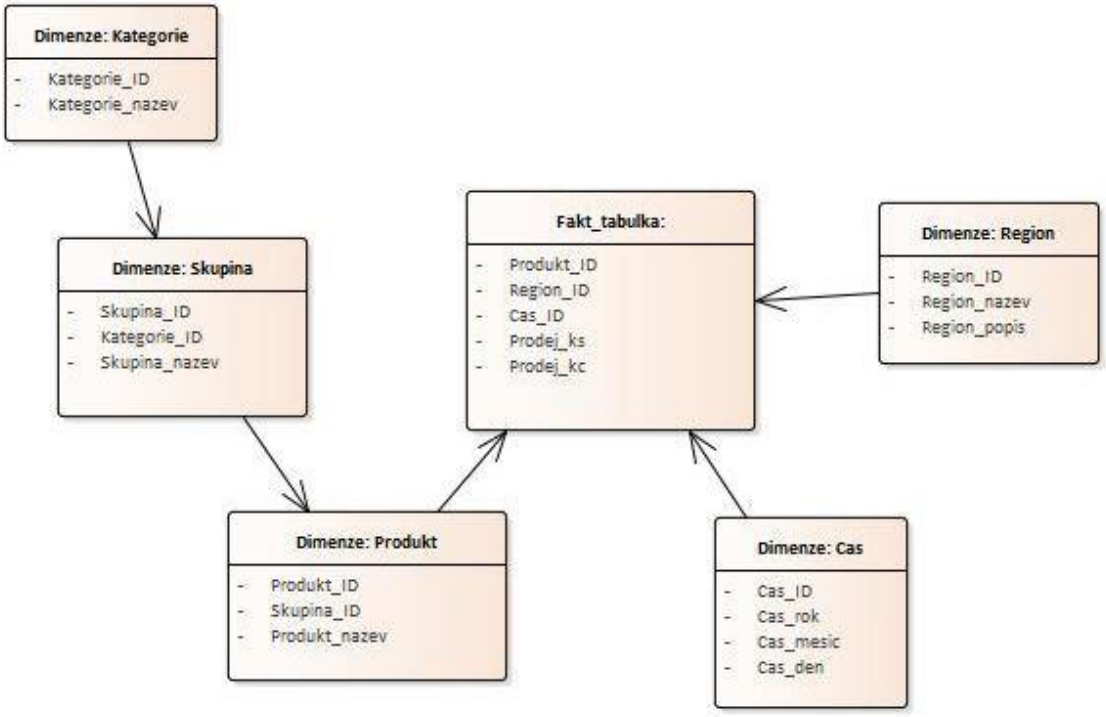
Schéma hvězdy neboli hvězdicové schéma je nečastějším používaným modelem pro převod relačních dat na dimenzionální. Grafická reprezentace modelu připomíná hvězdu, z toho důvodu se nazývá schéma hvězdy. V centru schématu se nachází tabulka faktů, na kterou jsou napojeny tabulky dimenzí. Každá dimenze zde představuje jednu tabulku, která může obsahovat několik atributů.



Obrázek 2: Hvězdicové schéma. Zdroj: (1).

Schéma sněhové vločky

Schéma sněhové vločky je rozšíření hvězdicového schématu. V tomto schématu je jako v hvězdicovém schématu v centru tabulka faktů, na kterou jsou napojeny tabulky dimenzí. Tabulky dimenzí jsou normalizované, tedy data se rozdělí do dalších tabulek dimenzí. Normalizováním tabulek se zmenší renundance dat, tím se ušetří diskový prostor. Na druhou stranu se ale zvýší počet tabulek dimenzí, což bude mít za následek nižší dotazovací výkon a menší účinnost analýzy dat.



Obrázek 3: Schéma sněhové vločky. Zdroj: (1).

2.2.1.2 Tabulka faktů

Tabulka faktů je centrální tabulka dimenzionálního modelu. Místo atributů obsahuje metriky neboli měřitelná data. Dále obsahuje primární klíč a cizí klíče tabulek dimenzí. Tabulka faktů obsahuje většinou nejvíce záznamů a jsou na ni napojeny dimenze.

Typy faktů

Fakta v tabulce můžeme rozdělit ještě podle použití. Máme tři základní rozdělení metrik (4):

- **aditivní** – metriky, které lze sčítat (agregovat) podle všech dimenzí;
- **semiaditivní –** metriky, které lze sčítat jen podle některých dimenzí;
- **neaditivní –** metriky, které nelze sčítat podle žádné dimenze, například podíl hodnot, procenta.

2.2.1.3 Tabulka dimenzí

Tabulka dimenzí je tabulka, která definuje metriky v tabulce faktů. Tabulka obsahuje atributy, které popisují danou dimenzi. Dále tabulka obsahuje primární klíč a může i obsahovat cizí klíč jiné dimenze. Atributy v dimenzionální tabulce mají hierarchickou stromovou strukturu. Příklad tabulky dimenze datum:

Datum:

*Rok
**Měsíc
***Den

Při návrhu tabulek dimenzí je důležité určit, jestli budou uchovávat historii dat. Každý atribut je nutné popsat a určit datový typ (číslo, text). Dále určit jakým způsobem daný atribut uchovává historii. Existují tři typy uchování historie:

- **Typ 1** – není požadována žádná historie;
- **Typ 2** – je požadována úplná historie;
- **Typ 3** – tento typ uchovává aktuální hodnotu atributu a v případě užití i první hodnotu;

Dimenze, ve které žádný atribut nevyžaduje ukládání historie, se nazývá dimenze typu 1. Jestliže jakýkoli atribut požaduje uchování historie, celá dimenze se nazývá dimenze typu 2. Tato dimenze se i označuje jako pomalu proměnná dimenze (anglicky – slowly changing dimension) (4). Pokud dimenze má jeden atribut nebo více typu 3 a ostatní typu 1, celá dimenze se nazývá jako typ 1 s atributem typu 3.

Typy dimenzí

Dimenze se dále dělí do několika kategorií.

Potvrzená dimenze

Tento typ je základní dimenzí v modelu. Tyto dimenze většinou patří do typu 2, protože je nezbytné uchovávat jejich historii. Mezi typické potvrzené dimenze patří Produkt, Zaměstnanec, Zákazník a další (4).

Dimenze data a času

Každá organizace potřebuje uchovávat data s konkrétním časem či datem. Například kdy byla prodaná určitá položka. Dimenze času se používá kvůli analýze, kde potřebujeme znát časový aspekt atributů. Struktura tabulky dimenze data a času záleží na organizaci. Některé organizace potřebují sledovat jednotlivé dny, jiným stačí uchovávat měsíce. Jestliže je potřeba sledovat a uchovávat čas, je lepší a přehlednější vytvořit novou dimenzi čas. Dimenze času by neměla být součástí dimenze datumu, protože by výrazně narostl objem řádků (4).

Degenerovaná dimenze

Degenerovaná dimenze je dimenze, která je zapsaná jako řádek v tabulce faktů a nemá vazbu na ostatní dimenze. Tato dimenze obsahuje malý počet atributů či dokonce jeden. Tato dimenze se zabalí jako součást primárního klíče do tabulky faktů (4).

Směsná dimenze

Směsná dimenze (anglicky junk dimension) se používá pro uložení atributů získaných z analýzy zdrojových dat, které nelze uložit do žádné dimenze, například indikátory. Tato dimenze by neměla obsahovat příliš mnoho atributů (4).

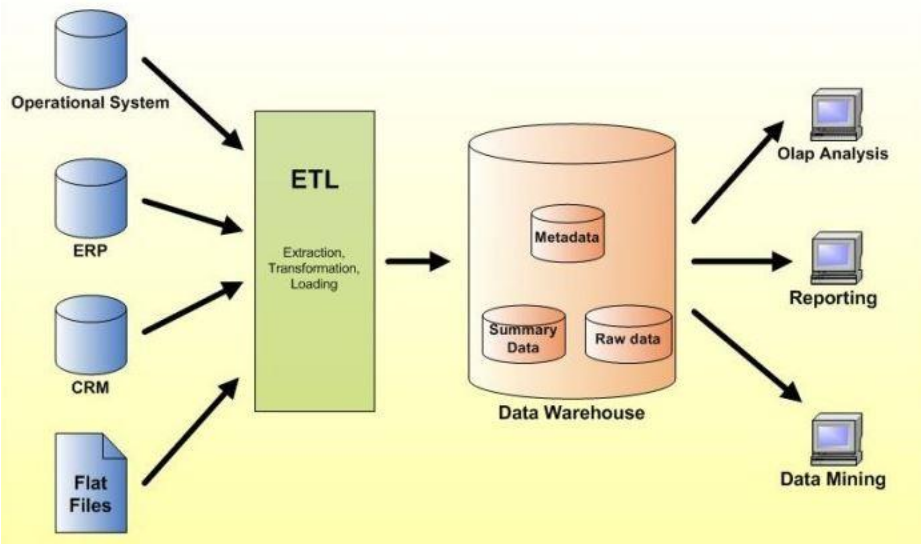
3 Datový sklad

Datový sklad (anglicky data warehouse) je systém, který umožnuje shromažďovat, organizovat, uchovávat a sdílet historická data. Termín „datový sklad“ použil poprvé Wiliam H. Inmon, který je širokou veřejností považován za otce myšlenky a návrhu datového skladu.

Definice datového skladu od Billa Inmona:

„Datový sklad je podnikově strukturovaný depozitář subjektově orientovaných, integrovaných, časově proměnlivých, historických dat použitých na získávání informací a podporu rozhodování. V datovém skladu jsou uložena atomická a sumární data.“ (5, str. 48).

Datový sklad je součástí business intelligence. Podnikoví uživatelé často využívají datový sklad k business inteligenci, která je založena na efektivním využití dat a následném uchování v datovém skladu. Na datové sklady můžeme přistupovat z různých pohledů. Jeden pohled vnímá datový sklad jako úložiště datového skladu s komponenty, jiný jako na kompletní systém ETL procesů a jednotlivých datových úložišť.



Obrázek 4: Datový sklad. Zdroj: (6).

Nyní si představíme pár pojmů, které byly dříve zmíněny.

- **ETL (Extract, Transform, Load) -** datový přenos, který extrahuje data ze zdrojové databáze, transformuje do cílové struktury a následně přenese data do cílové databáze.
- **Business intelligence**-je termín, který se vztahuje k procesům, technologiím, aplikacím a postupům, které zjednodušují podnikové rozhodování. Technologie pracuje s historickými daty a pomáhá přijímat podniková rozhodnutí (4).

3.1 Charakteristiky datového skladu

Jak bylo výše zmíněno, datový sklad je subjektově orientovaný, integrovaný, časově variabilní a neměnný. Právě těmito vlastnostmi se liší od provozní databáze. Data se získávají z provozních databází pomocí ETL procesů. ETL proces data načte z provozních databází, transformuje do požadované struktury a následně přenese do datového skladu. Následně v datovém skladu můžeme provádět analýzy, které slouží k podnikovému rozhodování. Procesem ETL se budeme detailně zabývat ve čtvrté kapitole.

3.1.1 Subjektová orientace

Data se do datového skladu ukládají podle jejich typu než podle toho, v jaké aplikaci byly vytvořeny. Při subjektové orientaci jsou data rozdělována podle subjektu, např. výrobek, zákazník. To znamená, že jsou data uložena pouze jednou v datovém skladu, na rozdíl od produkčních systémů, kde jsou data rozdělena podle toho, pro jakou aplikaci mají být použita (5).

3.1.2 Integrovanost

Data v datovém skladu musí být jednotná a integrovaná. Data týkající se jednoho předmětu se ukládají pouze jednou. Do datového skladu dostáváme data z nekonzistentních systémů, proto musí být před uložením upraveny a sjednoceny pod jednotnou terminologii. Data v datovém skladu musí být konzistentní, jinak ztrácí datový sklad význam (5).

3.1.3 Časová variabilita

V datovém skladu se ukládají data s časovou variabilitou, to znamená že uložená data mají časový aspekt. Z důvodu provádění analýz nad datovým skladem, je potřeba znát historii dat. Na rozdíl od operačních databází, kde se data ukládají za kratší časové období (dny, měsíce), v datovém skladu se uchovávají za delší období převážně několik let (5).

3.1.4 Neměnnost

Data do datového skladu získáváme z různých datových zdrojů. Jakmile jsou data uložená v datovém skladu, už je nelze měnit. V datovém skladu můžeme provádět převážně dvě operace. Ukládat nové údaje do datového skladu a umožnit přístupy k nim. Data nelze měnit ručně ani uživatelskými nástroji (5).

3.2 Metody budování datového skladu

Při vytváření datového skladu je důležité vybrat správnou metodu budování. Budování datového skladu je velice individuální. Výběr metody závisí na firmě, která musí brát v potaz svojí organizační a kulturní strukturu. Nejčastější metody budování datového skladu jsou metoda „velkého třesku“ a přírůstková metoda.

3.2.1 Metoda „velkého třesku“

Metoda „velkého třesku“ spočívá ve vytvoření datového skladu pomocí jediného projektu. Celková doba tvorby projektu by neměla být moc dlouhá, protože se mohou změnit technologie nebo požadavky uživatelů. Výhodou této metody je fakt, že celý projekt datového skladu je vytvořen ještě před realizací. Vytváření datového skladu je velice náročné a proměnlivé. Nevýhodou je velké riziko ze změny uživatelských požadavků. Metoda „velkého třesku“ se skládá ze tří fází (5).

- Analýza požadavků
- Vytvoření podnikového datového skladu
- Vytvoření přístupu buď přímo, nebo přes datové trhy

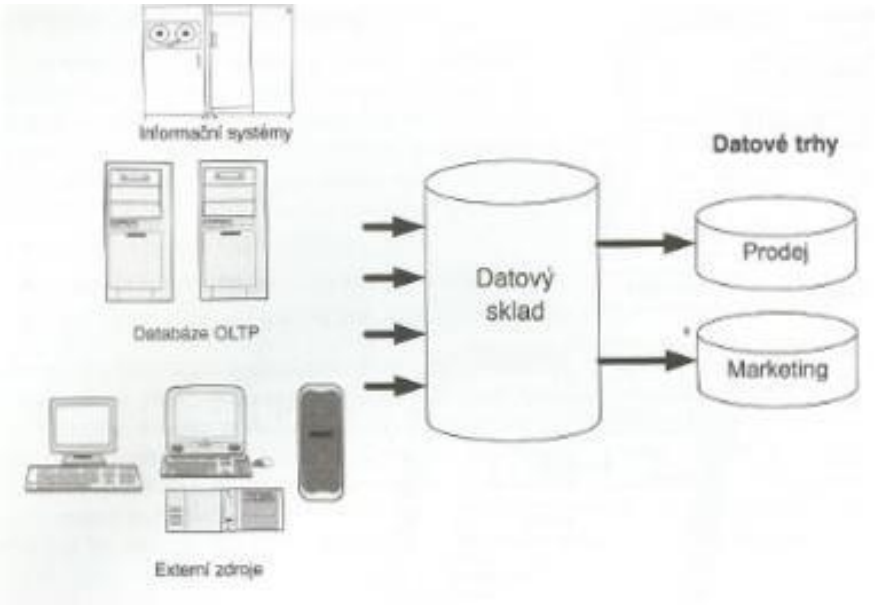
3.2.2 Přírůstková metoda

Tato metoda předpokládá budování datového skladu po jednotlivých částech, které následně zapadají do celkové architektury datového skladu. Vybudovanou část zasadíme do provozu a poskytneme koncovým uživatelům. Jakmile je část funkční a otestována koncovými uživateli, je možné vybudování další části. Tímto způsobem se pokračuje až do vybudování celého datového skladu. Přírůstková metoda má oproti metodě „velkého třesku“ mnoho výhod. Budováním datového skladu po částech zachovává soudržnost s uživatelskými požadavky. Dále umožňuje rozšíření architektury a zabezpečuje rychlejší návrat investic (5). Přírůstková metoda se realizuje ve dvou variantách:

- Přírůstková metoda „shora dolů“
- Přírůstková metoda „zdola nahoru“

3.2.2.1 Přírůstková metoda „shora dolů“

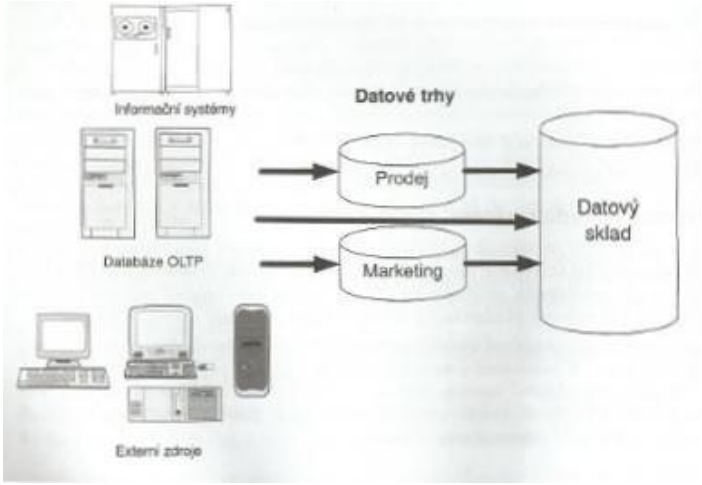
Tato metoda je poměrně rychlá na implementaci. Nejdříve vytvoříme konceptuální model datového skladu na základě uživatelských požadavků a současně stanovíme hierarchii předmětných oblastí. Následně postupně vytváříme datové trhy jednotlivých předmětných oblastí. Výhodou této metody je rychlá implementace a poměrně malá náročnost na analýzu. Nevýhodou metody „shora dolů“ jsou velké vstupní náklady.



Obrázek 5: Přírůstková metoda "shora dolů". Zdroj: (5).

3.2.2.2 Přírůstková metoda „zdola nahoru“

U této metody nejdříve budujeme datové trhy předmětných oblastí datového skladu a následně celkový konceptuální model datového skladu. Prioritu zde mají údaje před obchodním ziskem. U této metody vstupuje do popředí IT oddělení podniku, které se v mnoha podnicích nepovažuje za „lídra“ v oblasti marketingu, proto se o mnoha strategických záměrech dozvídá jako poslední (5). Z toho důvodu mohou navrhnout něco, co je vůči strategickým záměrům podniku neaktuální.



Obrázek 6: Přírůstková metoda "zdola nahoru". Zdroj: (5).

3.3 Architektura datového skladu

Architektura datového skladu popisuje návrh systému datového skladu, komponenty a jejich vzájemné vztahy. Návrh architektury závisí na organizační struktuře firmy. Důležité je si stanovit strategické cíle datového skladu v rámci organizace, které musí být v souladu s požadavky dané organizace. Dále definovat budoucí využití datového skladu a správu dat. Správa dat a budoucí použití závisí na správném výběru architektury. V této kapitole je čerpáno ze zdrojů (4)(7).

3.3.1 Vrstvy architektury

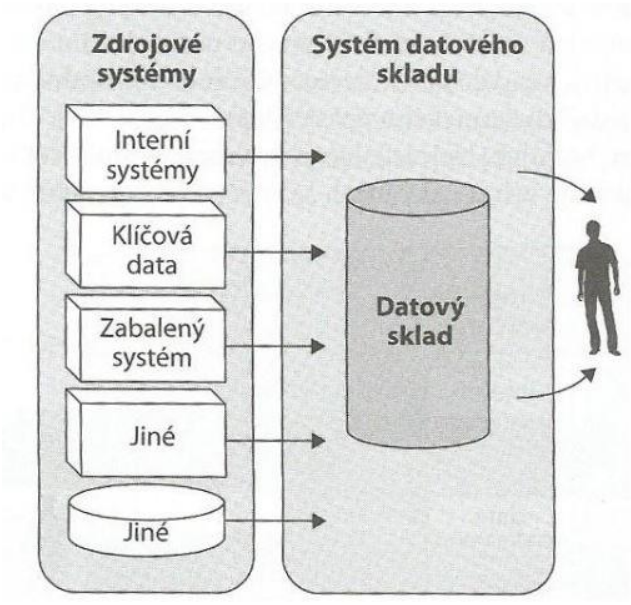
Systém datového skladu lze rozdělit na několik propojených systémů. Tyto jednotlivé systémy se označují vrstvy architektury. Jednotlivé architektury mají jiný počet vrstev. Ty nejběžnější si popíšeme v následujících kapitolách.

3.3.1.1 Architektura s jednou vrstvou

Architektura s jednou vrstvou nepatří mezi datové sklady, protože nerozlišuje transakční a analytické oblasti. Spíše se jedná o systém pro vykazování. Při analytickém dotazování dochází ke vstupu přímo do zdrojových systému, tím se zatíží podsystémy a může docházet k ovlivnění podnikových operací. V této architektuře nedochází k žádné redundanci dat, tímto nám odpadá proces ETL. Architektura s jednou vrstvou se používá pro on-line reporting.

3.3.1.2 Architektura se dvěma vrstvami

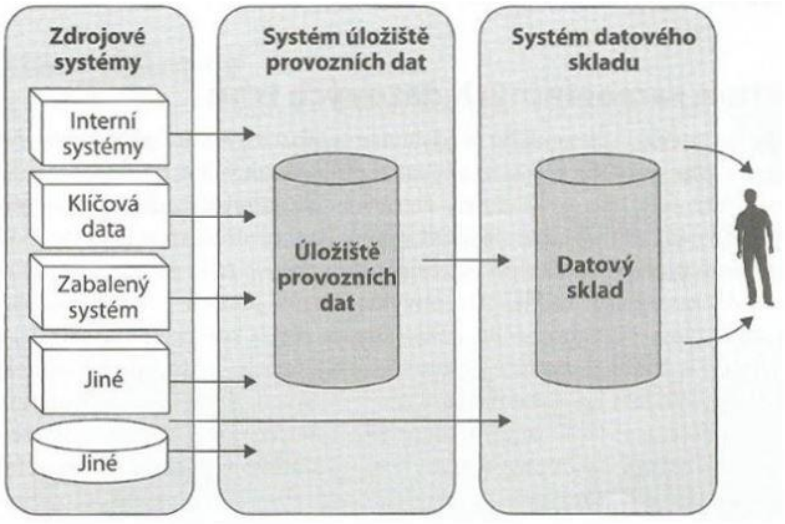
Architektura se dvěma vrstvami je nejběžnější architekturou datového skladu. Datový sklad se dvěma vrstvami zahrnuje systém datového skladu a zdrojové systémy. Systém datového skladu a zdrojové systémy jsou od sebe odděleny. Díky rozdělení těchto systémů je možné vykonávat analýzy datového skladu nezávisle na zdrojových systémech. Datový sklad se dvěma vrstvami má vlastní datový model, nezávislý na zdrojových systémech. Vytvoření nezávislého datového skladu nám umožňuje integraci dalších systémů do datových skladů.



Obrázek 7: Architektura se dvěma vrstvami. Zdroj: (4).

3.3.1.3 Architektura se třemi vrstvami

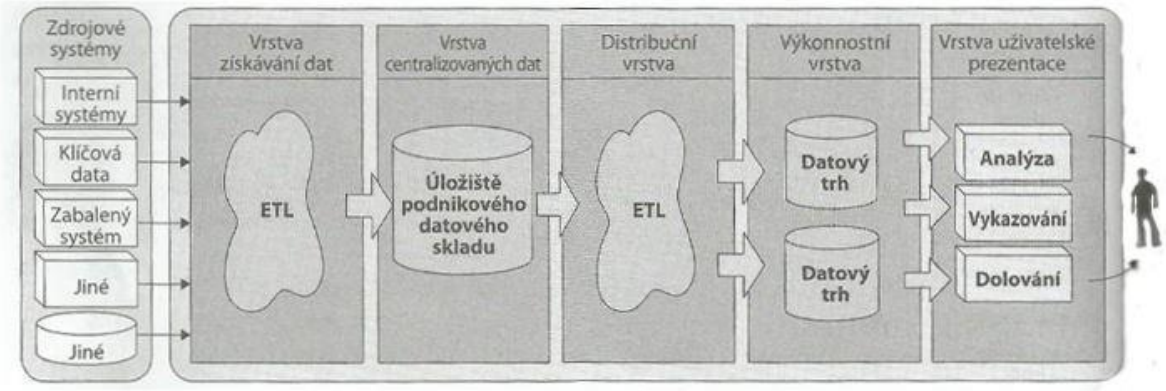
Architektura se třemi vrstvami obsahuje další systém, který bývá většinou úložiště provozních dat. Úložiště provozních dat (anglicky operational data store – ODS) získává data ze zdrojových systémů a následně data poskytuje datovému skladu. Data v ODS se uchovávají obvykle bez historie. ODS zahrnuje správu dat, což zjednodušuje integraci systému do datového skladu. ODS poskytuje aktuální pohled na data a často uchovává zákaznické informace (7). ODS se dále dělí na jednotlivé třídy podle rychlosti přenosu dat.



Obrázek 8: Architektura se třemi vrstvami. Zdroj: (4).

3.4 Architektura centrálního úložiště

Tato architektura byla navržena Billem Inmonem (4). Architektura centrálního úložiště je založena na přístupu „shora dolů“. Hlavní komponentou je centrální úložiště, ve které se ukládají normalizovaná data s plnou časovou historií. K datům se dále přistupuje pomocí datových trhů. Datové trhy jsou podmnožinou datového skladu ale nemusí být nutně normalizované. Datové trhy mohou být navrženy v 3NF, schématu sněhové vločky nebo hvězdicovém schématu, což nám dovoluje provádět analytické dotazy. Tento výběr závisí na požadavcích podniku. Úložiště zde představuje centrální zdroj dat. Podnikoví uživatelé přistupují k datům pomocí datových trhů, které mohou být zaměřeny jen na určitou oblast. Tato architektura se zaměřuje převážně na správu podnikových dat.



Obrázek 9: Architektura centrálního úložiště. Zdroj: (4).

Každá architektura má své výhody i nevýhody, proto se musí každý podnik při výběru architektury rozhodnout, pro jaký účel bude datový sklad využívat.

Výhody architektury centrálního úložiště:

- Díky normalizovaným datům v datovém skladu můžeme provádět odlišné analytické operace, než pro které byly určeny.
- Datový sklad podporuje tvorbu datových tržišť v dimenzionální formě i 3NF.
- Lze budovat téměř neomezeně datových tržišť pro konkrétní potřeby uživatelů (1).

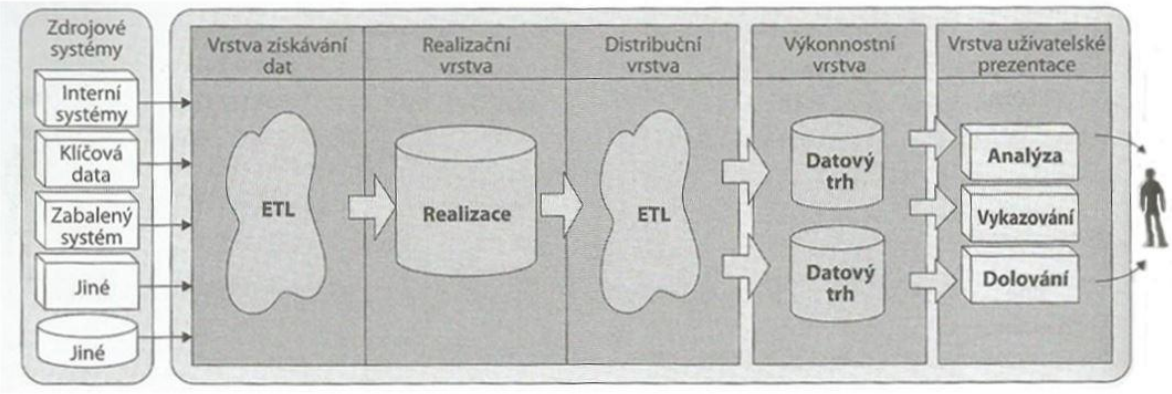
Nevýhody:

- Změna návrhu úložiště kvůli novým požadavkům ovlivní všechny systémy.
- Vývoj datového skladu s centrálním úložištěm trvá déle a je finančně náročnější.

3.5 Architektura sběrnice

Tato architektura byla navržena Ralphem Kimballem (4). Přístup orientovaný na datové trhy. Datové trhy mají návrh hvězdice nebo sněhové vločky. Tato architektura je více zaměřena na podnikovou analýzu. Celkový návrh datových tržišť tvoří tabulky faktů a potvrzené dimenze. Potvrzené dimenze jsou jednotné, znovupoužitelné dimenze v rámci datových trhů v celém podniku. Potvrzené dimenze se využívají z důvodu sjednocení terminologie a struktury mezi datovými trhy. Každé oddělení podniku může vytvářet svůj

vlastní datový trh. Všechna tato oddělení používají k analýze stejné dimenze, tímto se zjednoduší použití dat v podniku.



Obrázek 10: Architektura sběrnice. Zdroj: (4).

Výhody architektury sběrnice (1):

- Architektura využívá dimenzionální model, který se využívá pro analytické operace.
- Analytické potřeby jednotlivých oddělení jsou vyřizovány v nejrychlejším možném čase.
- Aplikace budované ohledně jednotlivých tržišť poskytují uživatelům informace potřebné pro analytickou činnost.

Nevýhody:

- Ve velkých podnicích je často složité vytvoření potvrzených dimenzí.
- Datové trhy se mohou překrývat, tím dochází k duplicitě ostatních komponent (8).
- V případě nutnosti použití relační databáze je potřeba vytvořit řešení mimo oblast datových tržišť (1).

4 ETL Proces

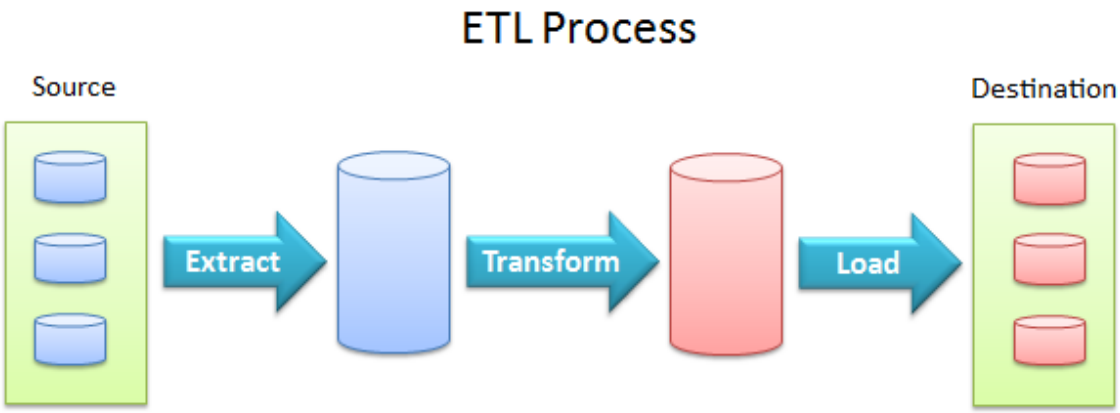
ETL procesy jsou nejdůležitější částí při budování datového skladu. Proces ETL se skládá ze tří fází (Extrakce, Transformace, Načítání). Data do datového skladu většinou získáváme z různých nesourodých zdrojů. Proces ETL extrahuje data ze zdrojových systémů, transformuje do požadované struktury a následně nahraje do datového skladu. Název ETL se skládá z počátečních písmen jednotlivých fází (Extract, Transform, Load). Proces ETL se běžně označuje jako datová pumpa. Jednotlivé fáze ETL procesu často probíhají paralelně.

ETL vs ELT

ELT (Extract, Load, Transform) proces oproti ETL nejprve extrahuje data, načte data do databáze a poté probíhají jejich transformace. ELT se používá spíše na větší objem dat a potřebuje dostatečný výkon pro transformace dat v cílové databázi. V procesu ELT odpadá transformační modul, protože transformace probíhají v cílové databázi.

Základní úlohy ETL procesu

- Odstraňuje chyby a opravuje chybějící data
- Zachycuje tok transakčních dat pro úschovu
- Upravuje data z více zdrojů k jednotnému použití
- Poskytuje datům hodnotu a upravuje pro koncové uživatele (9).



Obrázek 11: Proces ETL. Zdroj: (10).

4.1 Fáze extrakce

První fáze ETL procesu zahrnuje selekci a extrakci dat ze zdrojových systémů. Zdrojové systémy bývají velmi různorodé, mohou být umístěny v odlišných operačních systémech (Windows, Linux, Mac), databázových systémech (MS SQL Server, Oracle) či v jiných formátech. Úlohou extrakce je získání údajů právě z těchto zdrojů. Zní to jednoduše, ale fáze extrakce je nejdůležitější fází ETL procesu. Při nesprávné extrakci může celý ETL proces selhat, protože ostatní fáze jsou na extrakci závislé. Z toho důvodu se po extrakci kontrolují zpětně data v cílové databázi vůči zdrojové. Fáze extrakce bývá nejnáročnější fází ETL procesu, co se výkonu týče, proto často bývá optimalizována vypnutím různých podpůrných procesů.

Nejčastějším zdrojem jsou relační databáze. Mezi další zdroje patří například „flat files“. „Flat files“ jsou tabulkové databáze tvořené jedním souborem. Mohou být v textové podobě, ale i binární.

4.2 Fáze transformace

Druhá fáze transformace se skládá z několika dalších úloh. První úlohou je tzv. čištění nečistých dat. Nečistá data jsou data, která obsahují závažné chyby. Čištění dat zajišťuje maximální možnou kvalitu dat, ještě před transformacemi. Transformace dat má co nejlépe připravit data před načtením do úložiště. Při samotné transformaci řešíme několik problémů.

Jedním z problému jsou chybějící hodnoty (NULL hodnoty) a duplicitní záznamy. Duplicitní záznamy jsou totožné záznamy uložené v databázi více než jednou. Při transformaci se mohou potom tyto hodnoty promazat. Větší problém je s chybějícími hodnotami. U menšího objemu dat můžeme tyto hodnoty ignorovat. Dále můžeme chybějící hodnoty doplnit z jiných zdrojů nebo dočasně ponechat v OLTP systému a zpracovat později. Dalšími problémy jsou nejednoznačné údaje, odlišné názvy objektů a pojmů, různé peněžní měny, jiné formáty čísel a textových řetězců, referenční integrita nebo chybějící datum. Referenční integrita jsou hodnoty popisující vztahy mezi tabulkami. Například hodnota v tabulce A musí odkazovat na hodnotu v tabulce B. Hodnota v tabulce A je poté získávána podle tabulky B. Tato integrita je v databázovém systému zajištěna pomocí cizího klíče. Transformace bývá vyjádřená v jazyce SQL nebo v „pseudokódu“.

4.3 Fáze načítání

Fáze načítání je poslední fází ETL procesu. V této fázi dochází k přenosu dat z paměti zdrojových dat nebo z přechodného úložiště do datového skladu. Data se přenesou a uloží do databázových tabulek dle formátu datového skladu (tabulky dimenzí a faktů). Celý tento proces by měl být plánovaný a plně automatizovaný. Při prvotním načítání do datového skladu může jít o obrovské množství dat. Dále dochází k načítání v určitých časových intervalech.

K neúplným datům může dojít například, když při načítání dat do datového skladu probíhá zároveň čtení dat z datového skladu. Po načtení údajů většinou probíhá jejich indexování, z důvodu optimalizovaného přístupu k datům (5).

Zpravidla máme tři typy načítání:

- **Počáteční** – označuje první načítání dat do datového skladu.
- **Obnovení** – odstraňuje existující data a opakuje načtení příslušných dat.
- **Aktualizace** – načítání nových dat do datového skladu.

4.4 Nástroje ETL

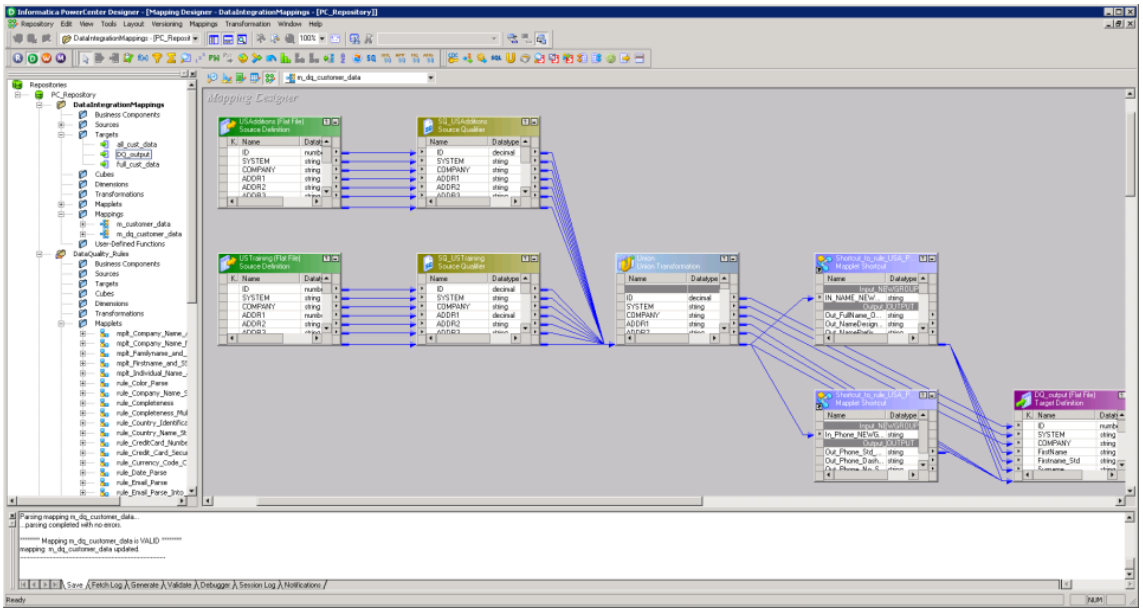
ETL nástroje pomáhají s tvořením ETL procesů a celkově při práci s daty. Dříve se vytvářeli ETL procesy pomocí dlouhých SQL kódů. V dnešní době existují nástroje, které práci s ETL procesy usnadňují a urychlují. ETL nástrojů je celá řada. V této práci si popíšeme pár základních. Mezi ETL nástroje patří:

- Informatica PowerCenter
- Oracle Data Integrator
- MS SQL Server Integration Services
- SAP Data Services

4.4.1 Informatica PowerCenter

PowerCenter je velmi rozšířený a osvědčený nástroj. Nástroj slouží pro tvorbu ETL/ELT procesů, profilování dat, různé čištění dat nebo migraci dat. Tento nástroj je schopný zpracovat data z mnoha zdrojů (databáze, soubory, aplikace). PowerCenter nástroj obsahuje skvělý managment s metadaty (11). Nástroj automaticky vytváří metadata, která lze následně spravovat v aplikaci Metadata Manager (specializovaná aplikace Informatica) či jiných reportovacích nástrojích.

Tento nástroj je placený. Při výběru tohoto nástroje na tvorbu ETL procesů je nutno počítat s větší počáteční investicí. Výhodou nástroje je snadná manipulace s daty.



Obrázek 12: Ukázka nástroje Informatica PowerCenter. Zdroj (12).

4.4.2 MS SQL Server Integration Services

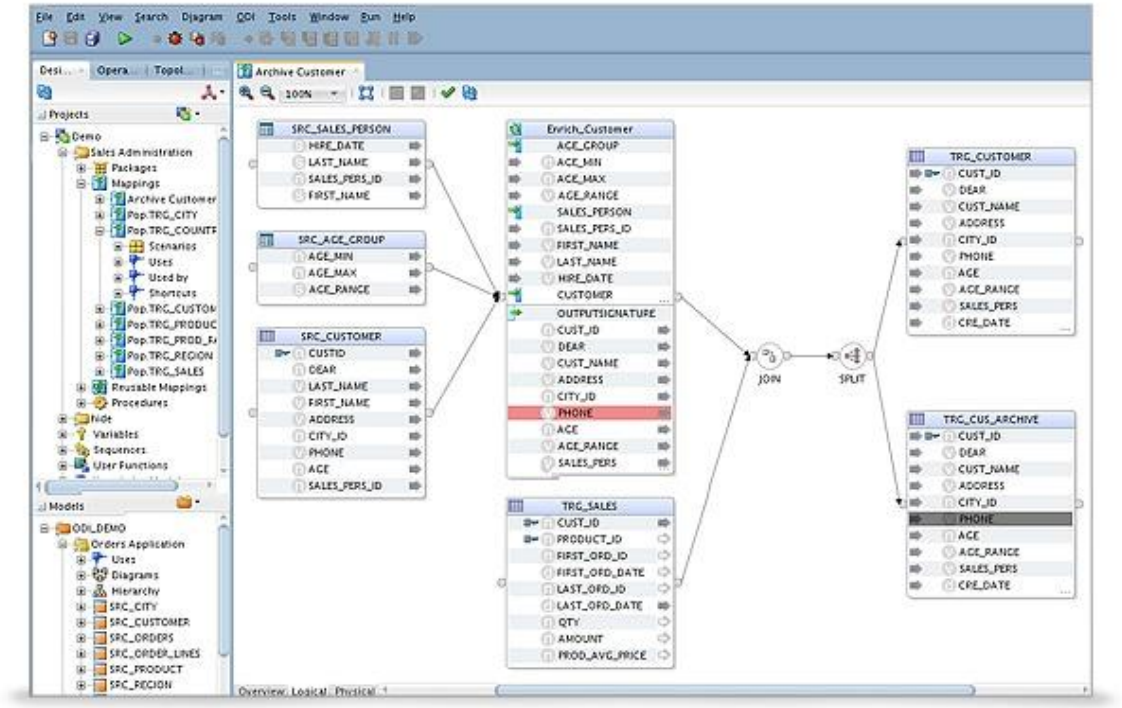
MS SQL Server Integration Services (SSIS) je nástroj od společnosti Microsoft. SSIS řeší datovou integraci mezi jednotlivými systémy. Zdrojem i cílem mohou být relační i nerelační databáze, webové služby, textové soubory nebo excelové soubory. Pro tvorbu ETL procesů, datových integrací slouží grafické prostředí Visual Studio, kde lze celý proces namodelovat a následně nahrát na MS SQL server. Namodelovaný proces dále

využívá prostředků serveru. Grafické prostředí obsahuje i debugger pomocí kterého lze namodelovaný proces zkontrolovat ještě před nahráním na server.

V případě potřeby využití jiného zdroje, než které jsou naprogramované ve Visual Studiu, lze datový konektor naprogramovat v MS.NET Frameworku.

4.4.3 Oracle Data Integrator

Oracle Data Integrator je nástroj pro integraci dat a tvorbu ETL/ELT procesů od společnosti Oracle. Data Integrator (ODI) také disponuje grafickým prostředím pro přehledné modelování. Tento nástroj pokrývá všechny požadavky na integraci dat.



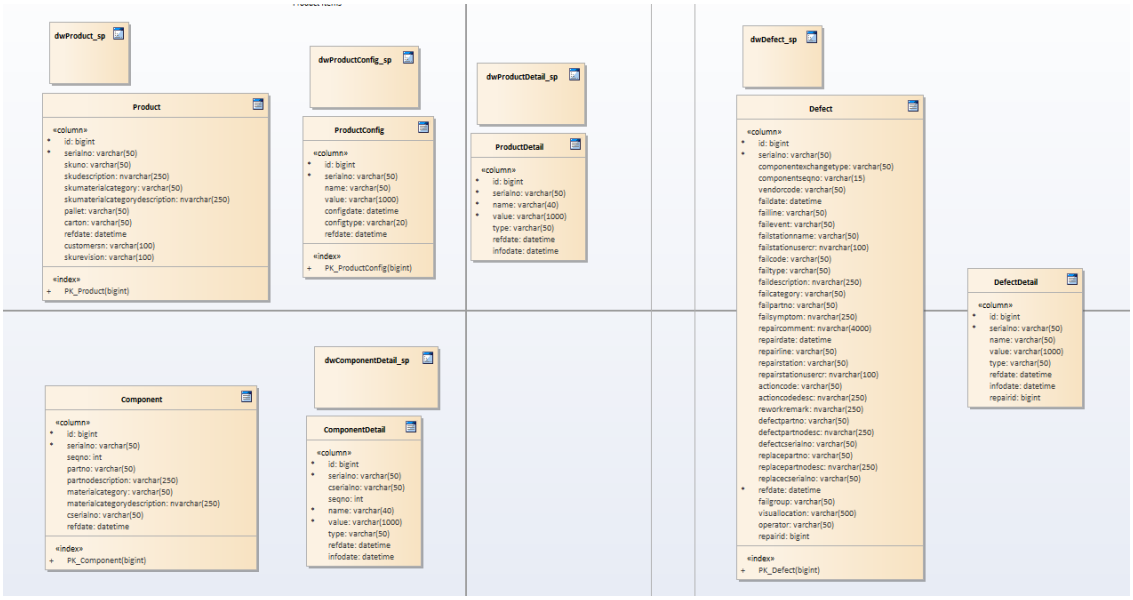
Obrázek 13: Ukázka grafického prostředí Oracle Data Integrator. Zdroj (13).

5 Analýza datového modelu datového skladu

Data z výroby se uloží do provozních databází a potom pomocí ETL procesu se transformují a uloží do datového skladu. Datový model datového skladu spolupracující společnosti je rozdělen do několika částí.

5.1 Popis datového modelu datového skladu

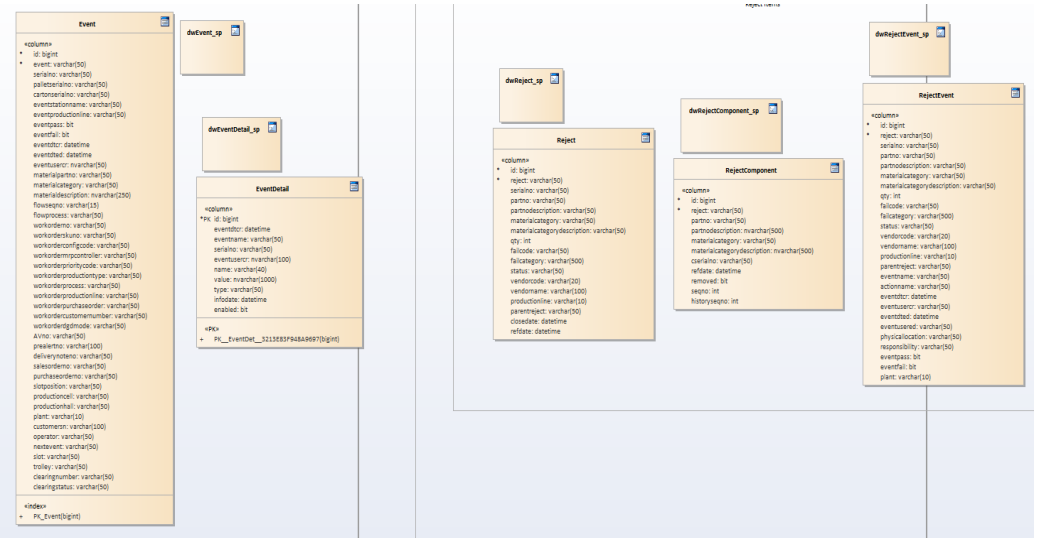
První část je zaměřena na produkt a jeho komponenty. Jsou zde tabulky s názvy Product, ProductConfig, ProductDetail, Component, ComponentDetail, Defect a DefectDetail. Do tabulky Product se ukládají data o vyrobeném produktu. Tabulka má primární klíč ID typu bigint. Dále má povinný atribut serialno (seriové číslo) typu varchar. Tabulka Product obsahuje řadu dalších atributů popisující produkt např. refdate, pallet, carton nebo skudescription. Tabulka ProductDetial slouží pro ukládání více detailních dat o produktu. Pro zaznamenávání změn produktu slouží tabulka ProductConfig. Atributy této tabulky jsou configtype, configdate, refdate, name, value a povinné atributy id a serialno. Tabulka Component slouží pro ukládání dat z výroby jednotlivých komponent produktu. Tabulka Defect a DefectDetail slouží k ukládání chyb, které nastanou při výrobě produktu.



Obrázek 14: Ukázka první části modelu. Zdroj (vlastní).

Další část je zaměřena na proces výroby. Tato část obsahuje tabulky Event, EventDetail, Reject, RejectComponent, RejectEvent. Tabulky Event a EventDetail uchovávají data z pracovního procesu výroby produktu. Příklad: produkt jede po lince a

u každého procesu (eventu) se namontuje nová komponenta produktu. Data z jednotlivého procesu se upraví pomocí ETL procesu a následně uloží do tabulky Event a EventDetail. Tabulky Reject, RejectComponent a RejectEvent jsou podobné jako tabulka Defect. Tyto tabulky uchovávají data o chybách při výrobním procesu, které nastanou při chybném procesu nebo při chybné komponentě.



Obrázek 15: Ukázka druhé části modelu. Zdroj (vlastní).

Třetí část je zaměřena na pracovní plán. Tato část obsahuje mnoho tabulek pro uložení potřebných dat z provozních databází o pracovním plánu. Jsou zde tři hlavní tabulky s názvy Workorder, WorkOrderOnDownload a WorkOrderOnStart, které ukládají data o pracovním plánu. Data pracovního plánu se mohou změnit, z toho důvodu se ukládají při stažení pracovního plánu i při začátku výroby. Dále jsou tam tabulky WorkOrderItemHistory, WorkOrderOnDownloadHistory a WorkOrderOnStartHistory, které uchovávají historická data o pracovních plánech. Tabulky s názvy WorkOrderOperation, WorkOrderOnDownloadOperation a WorkOrderOnStart-Operation ukládají data o jednotlivých operacích. Tyto tabulky mají většinu atributů časového aspektu. Tyto tři části jsou hlavními částmi celého datového modelu.

Dále je tu část, která je zaměřena na jedinou oblast výroby. Obsahuje tabulky s názvy DeviceState, DeviceStateMaterial, DeviceSignal, MaterialMovementOriginalLocaiton a MaterialMovement. Datový model obsahuje doplňující tabulky s názvy ShiftPlan, ProductionStandart, ProductStructure, Version, VersionProduct a mdMarketManagment.

5.2 Analýza datového modelu

Datový model spolupracující společnosti spíše představuje centrální úložiště nežli model datového skladu. Model obsahuje tabulky faktů, ale žádné dimenze. Mezi tabulkami nejsou žádné vztahy kromě jedné oblasti. Model nelze přirovnat k hvězdicovému ani vločkovému modelu datového skladu. Nicméně lze s daty v tomto modelu provádět analytické operace, takže se model zdánlivě podobá datovému skladu. Každá firma si může vytvořit datový sklad podle svých potřeb, žádný model není univerzální pro všechny. V datovém modelu datového skladu by měly být dimenze a jedna tabulka faktů a vztahy mezi nimi. Tabulek faktů ale může být mnohem více, vše záleží na potřebách a požadavcích firmy pro kterou je model určen. Tabulka faktů by měla mít vztahy na všechny dimenze a obsahovat cizí klíče dimenzí. Tento model je postaven na analýzu chyb při výrobě produktů a její objem za určitá časová období.

Navrhované tabulky dimenzí:

- Produkt
- Komponenta
- Datum a čas
- Pracovní linka
- Pracovník
- Pracovní plán
- „Výrobní chyba“

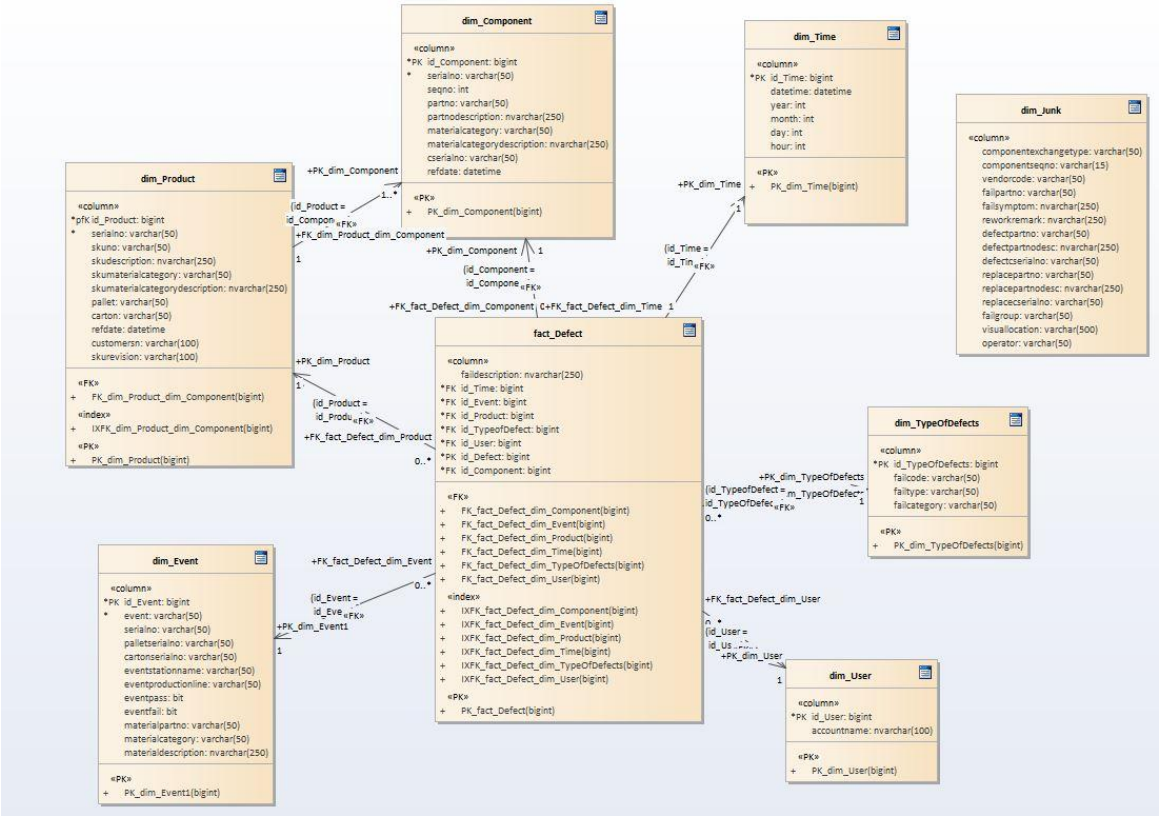
Tabulky faktů jsou potom zaměřeny na chyby ve výrobním procesu i při vyrobeném produktu.

5.3 Návrh datového modelu

Návrh je zaměřen na část celkového datového modelu, a to na chyby produktu. Při návrhu budeme vycházet z tabulky Defect. Pomocí analýzy dat jednotlivých atributů tabulky Defect je vytvořen základní model složen z dimenzí a tabulek faktů. Jednotlivé atributy tabulky Defect jsou rozděleny do několika dimenzí, které mají vztah na tabulku faktů. Při vytváření dimenzí a jejich atributů byly zkoumány jejich vztahy na jiné tabulky a analyzována korelace dat mezi jednotlivými atributy. Model je rozdělen na dvě části, jedna část je zaměřená na chyby a druhá na opravu chyb. Obě tyto části mají sdílené

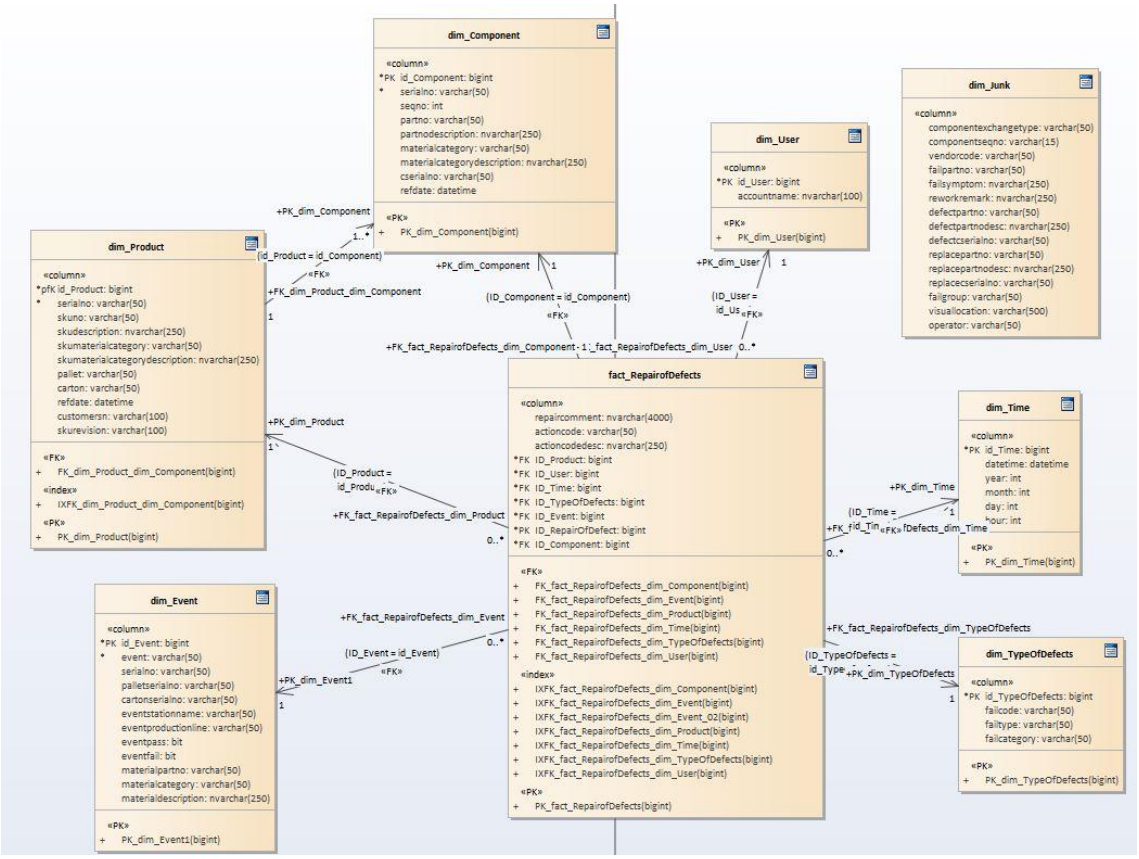
tabulky dimenzí. Model obsahuje dvě tabulky faktů s názvy Defect a RepairOfDefect. Dále je model složen z několika dimenzí s názvy Product, Component, User, Time, Event, TypeOfDefect a Junk. Dimenze obsahují jednotlivé atributy tabulky Defect, podle toho, co popisují. Dále má každá dimenze svůj primární klíč. Tabulky faktů obsahují atributy, primární klíč a cizí klíče všech dimenzí, které mají vztah s tabulkou. Navržený model je hvězdicového schématu.

První část modelu je zaměřena na chyby produktu. V této části jsou dimenze napojeny na tabulku faktů s názvem Defect, která obsahuje informace o chybě a cizí klíče všech dimenzí. Tabulky Product, Component a Event jsou převzaté z původního modelu. U tabulky Event byly upraveny a změněny některé atributy z původního modelu abychom vytvořili potřebnou dimenzi. Dimenze TypeOfDefect a User byly vytvořeny z tabulky Defect. Dimenze TypeOfDefect obsahuje atributy, které popisují chybu. Dimenze User zahrnuje informace o uživateli. Atributy dimenzí TypeOfDefect a User byly dříve obsaženy v tabulce Defect. Dále je v modelu dimenze Time, protože v datovém skladu se uchovává i historie. Tato dimenze má atributy popisující datum. Dimenze „Junk“ obsahuje atributy tabulky Defect, které nelze přiřadit k žádné dimenzi nebo neobsahují žádné data.



Obrázek 16: Část modelu zaměřená na chyby. Zdroj (vlastní).

Druhá část je zaměřena na opravu chyb. V této části je pouze jiná tabulka faktů, která je zaměřena na opravu chyb. Všechny dimenze jsou v modelu sdílené tudíž veškeré vztahy mezi tabulkami a dimenze jsou totožné s první částí.



Obrázek 17: Část zaměřená na opravu chyb. Zdroj (vlastní).

6 Návrh ETL procesu

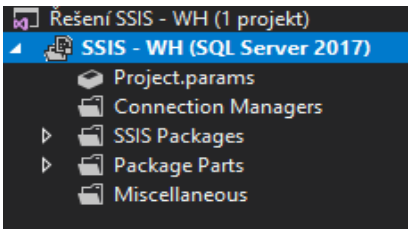
Procesy ETL spolupracující společnosti jsou psány ručně pomocí SQL jazyka. V této práci je použit na návrh ETL procesu nástroj MS SQL Server Integration Services (SSIS). Procesy jsou modelovány v programu Microsoft Visual Studio a následně nahrány do MS SQL Serveru. Data získaná od spolupracující společnosti jsou čistě anonymizovaná.

6.1 Návrh prvního ETL procesu

Tento proces je navrhnut jako zálohovací proces, který data převede z jedné tabulky do druhé. V tomto procesu dochází k záloze dat tabulek Product, Component, Workorder, Defect a Event. Jednotlivé tabulky byly popsány v kapitole 5.

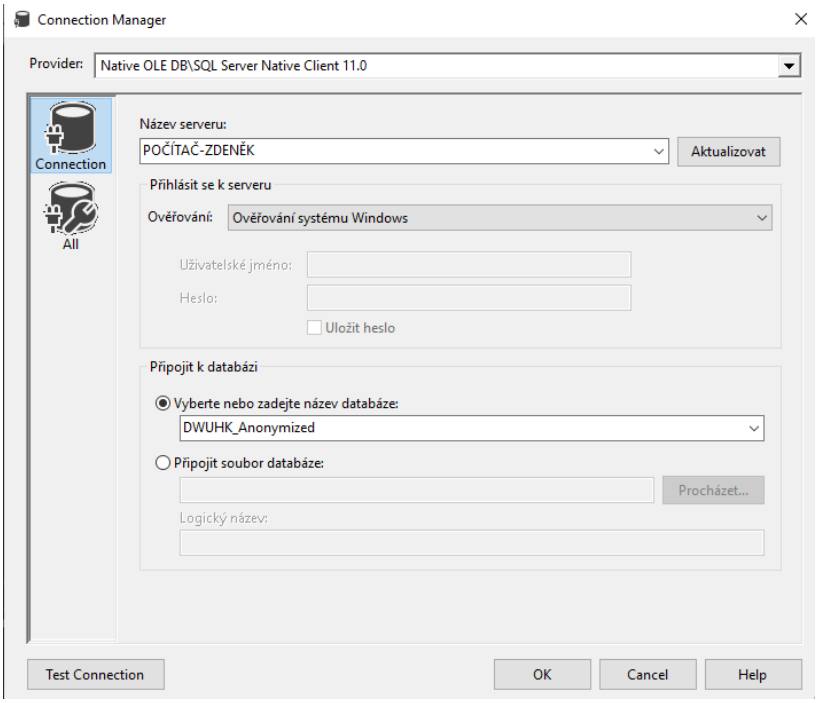
6.1.1 Postup implementace procesu

Nejdříve byl vytvořen nový projekt v Microsoft Visual Studio s rozšířením Business Inteligence a vybereme Integration Services Project.



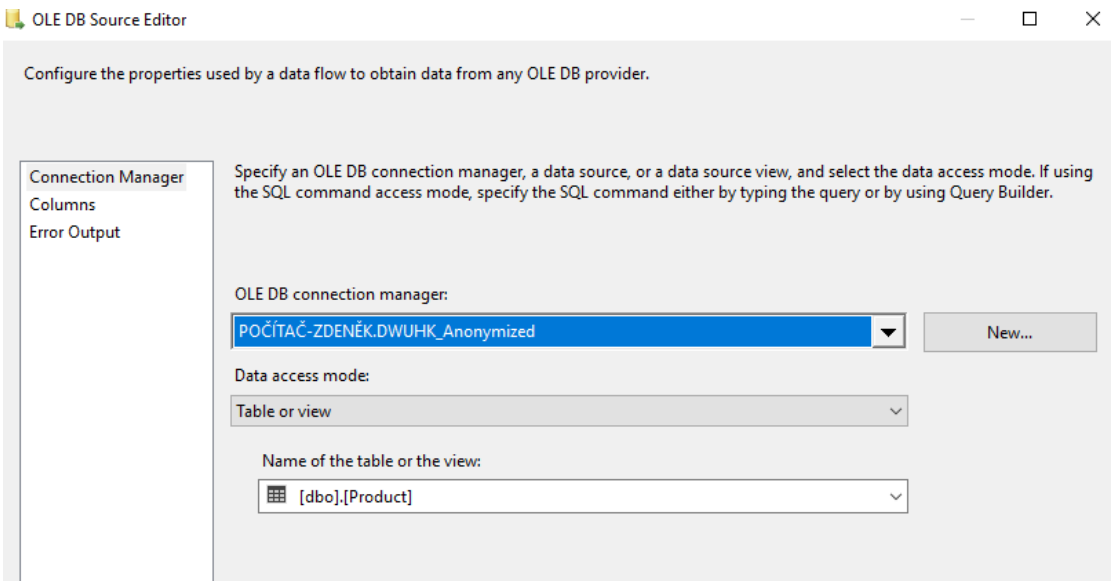
Obrázek 18: Průzkumník řešení. Zdroj (vlastní).

Na obrázku je prázdný průzkumník řešení, ve kterém lze začít pracovat. Nejprve je potřeba vytvořit datový zdroj. Na výběr je široké množství datových zdrojů od textového, html, excel souboru nebo databáze. V tomto řešení je použita databáze se jménem DWUHK_Anonymized, která je uložena na serveru POČÍTAČ-ZDENĚK. Potvrzením první záložky je tato databáze připojena.



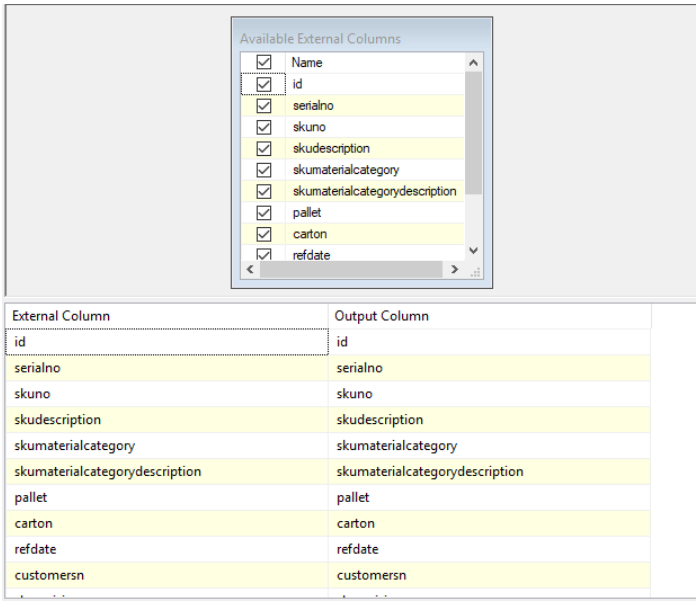
Obrázek 19: Připojení databáze k řešení. Zdroj (vlastní).

Po vytvoření zdroje byl založen SSIS balíček s názvem BackUpDatabase, protože bude modelován zálohovací proces. V panelu Control Flow je vybrán z nástrojů Data Flow Task, přetažen na pracovní plochu a následně pojmenován Extract and Load. Rozkliknutím panelu se zobrazí záložka Data Flow a lze začít modelovat. V záložce Data Flow z panelu nástrojů je vybrán Source Assistant, pomocí kterého je vybrána zdrojová tabulka.



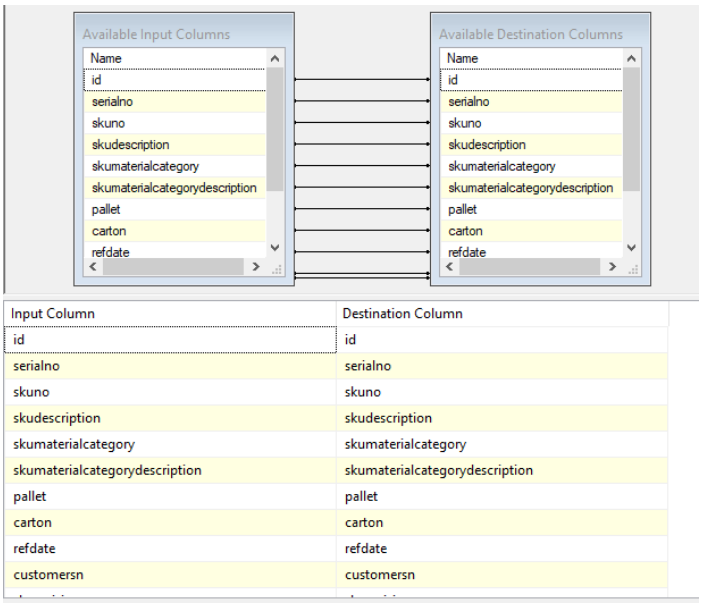
Obrázek 20: Výběr zdrojové tabulky. Zdroj (vlastní).

V záložce Columns lze zvolit, které atributy budou použity. V tomto procesu jsou vybrány k záloze všechny atributy.



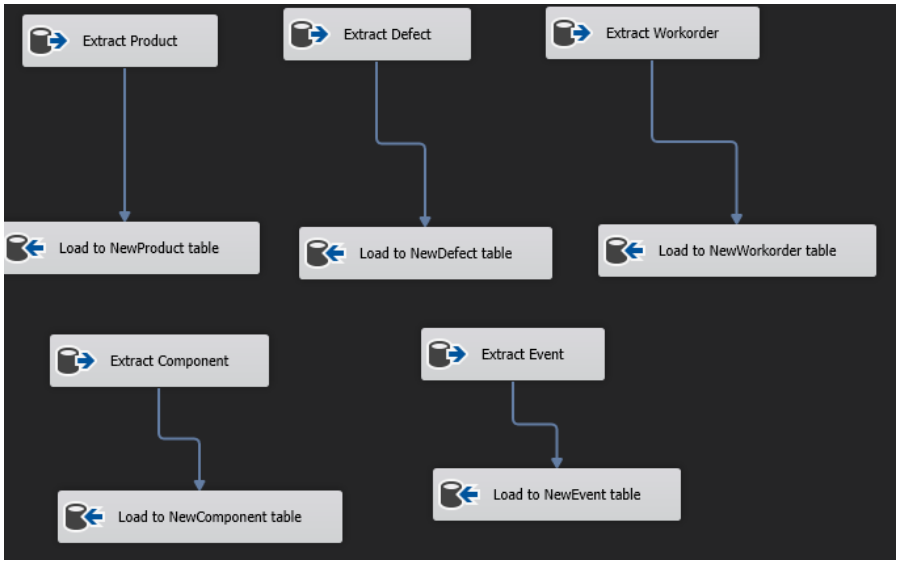
Obrázek 21: Přehled atributů tabulky Product. Zdroj (vlastní).

V dalším kroku je vytvořena destinaci, kam je potřeba data přenést nebo zálohovat. Pomocí podobného asistenta jako v předchozím kroku je použit Destination Assistant. V tomto asistentu je vytvořena destinační tabulka, kam budou data přenesena. V záložce Mappings je nutno správně namapovat atributy.



Obrázek 22: Mapování atributů tabulek Product. Zdroj (vlastní).

Tento postup je proveden na všechny vybrané tabulky, ze kterého dostaneme následující model. Po provedení tohoto procesu se data zkopírují a přenesou do nových tabulek.



Obrázek 23: Návrh celého modelu. Zdroj (vlastní).

Dalším rozšířením toho procesu je vytvoření přístupů tzv. loggů. Tyto logy se ukládají do samostatného textového souboru. Soubor obsahuje informace o provedeném procesu. Obsahuje název procesu, čas začátku a ukončení procesu, název počítače, ze kterého byl proces spuštěn, zdroj dat a následnou destinaci. Na stejný princip je vytvořen textový soubor, který uchovává informace o chybách/error, které se vyskytnou během procesu. Následně se tento balíček nahraje do MS SQL Serveru, konkrétně do SSIS katalogu, ke kterému už lze přistupovat přes Microsoft SQL Server Management Studio 2017.

id	serialno	skuno	skudescrip…	skumateria…	skumateria…	pallet	carton	refdate	customersn
1	SN000007…	SKU00000…	SKUD001…	MC0032	SKUMD0052	PALLET00…	NULL	31.01.2018 10:06:37	NULL
2	SN000000…	SKU00000…	SKUD001…	MC0032	SKUMD0052	PALLET00…	NULL	31.01.2018 10:12:56	NULL
3	SN000000…	SKU00000…	SKUD001…	MC0032	SKUMD0052	PALLET00…	NULL	31.01.2018 11:19:34	NULL
4	SN000008…	SKU00000…	SKUD001…	MC0032	SKUMD0052	PALLET00…	NULL	31.01.2018 11:35:52	NULL
5	SN000004…	SKU00000…	SKUD001…	MC0032	SKUMD0052	PALLET00…	NULL	31.01.2018 11:57:40	NULL
6	SN000009…	SKU00000…	SKUD001…	MC0032	SKUMD0052	PALLET00…	NULL	01.02.2018 3:44:32	CUSTSN0…
7	SN000002…	SKU00000…	SKUD005…	MC0032	SKUMD0052	PALLET00…	NULL	06.11.2017 12:08:01	NULL
8	SN000009…	SKU00000…	SKUD005…	MC0032	SKUMD0052	PALLET00…	NULL	06.11.2017 12:08:26	NULL
9	SN000003…	SKU00000…	SKUD005…	MC0032	SKUMD0052	PALLET00…	NULL	06.11.2017 13:07:31	NULL
10	SN000009…	SKU00000…	SKUD005…	MC0032	SKUMD0052	PALLET00…	NULL	06.11.2017 18:31:29	NULL
11	SN000001…	SKU00000…	SKUD005…	MC0032	SKUMD0052	PALLET00…	NULL	06.11.2017 18:34:49	NULL
12	SN000001…	SKU00000…	SKUD005…	MC0032	SKUMD0052	PALLET00…	NULL	06.11.2017 18:36:32	NULL
13	SN000010…	SKU00000…	SKUD005…	MC0032	SKUMD0052	PALLET00…	NULL	06.11.2017 12:07:39	NULL
14	SN000010…	SKU00000…	SKUD005…	MC0032	SKUMD0052	PALLET00…	NULL	06.11.2017 18:30:45	NULL
15	SN000011…	SKU00000…	SKUD001…	MC0032	SKUMD0052	PALLET00…	NULL	31.01.2018 11:38:31	NULL
16	SN000011…	SKU00000…	SKUD001…	MC0032	SKUMD0052	PALLET00…	NULL	31.01.2018 11:44:04	NULL
17	SN000012…	SKU00000…	SKUD001…	MC0032	SKUMD0052	PALLET00…	NULL	31.01.2018 12:01:53	NULL
18	SN000011…	SKU00000…	SKUD001…	MC0032	SKUMD0052	PALLET00…	NULL	31.01.2018 12:03:36	NULL
19	SN000003…	SKU00000…	SKUD001…	MC0032	SKUMD0052	PALLET00…	NULL	31.01.2018 9:49:36	NULL
20	SN000008…	SKU00000…	SKUD001…	MC0032	SKUMD0052	PALLET00…	NULL	31.01.2018 10:00:02	NULL
21	SN000007…	SKU00000…	SKUD001…	MC0032	SKUMD0052	PALLET00…	NULL	31.01.2018 10:09:04	NULL
22	SN000003…	SKU00000…	SKUD002…	MC0032	SKUMD0052	PALLET00…	NULL	02.02.2018 7:02:37	NULL
23	SN000010…	SKU00000…	SKUD005…	MC0032	SKUMD0052	PALLET00…	NULL	06.11.2017 18:50:26	NULL
24	SN000009…	SKU00000…	SKUD005…	MC0032	SKUMD0052	PALLET00…	NULL	06.11.2017 19:06:02	NULL
25	SN000009…	SKU00000…	SKUD005…	MC0032	SKUMD0052	PALLET00…	NULL	06.11.2017 19:41:58	NULL
26	SN000010…	SKU00000…	SKUD005…	MC0032	SKUMD0052	PALLET00…	NULL	07.11.2017 0:31:38	NULL

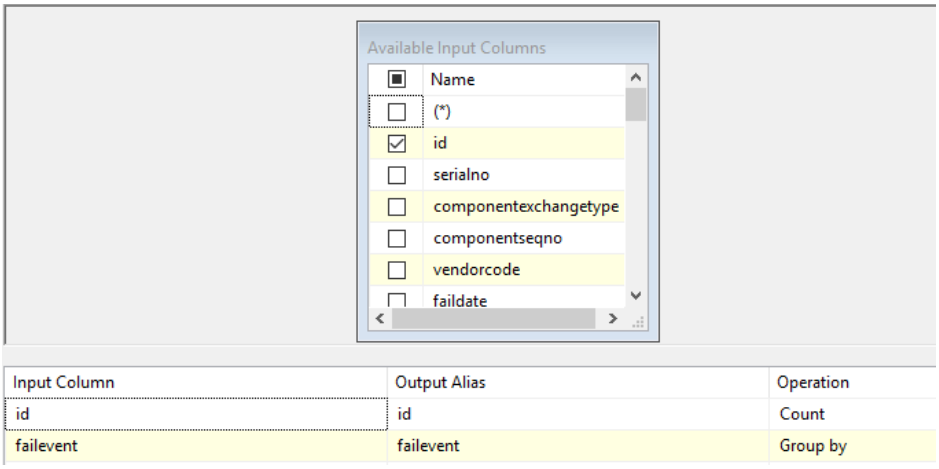
Obrázek 24: Pohled na data v nové tabulce Product. Zdroj (vlastní).

6.2 Návrh druhého ETL procesu

Druhý proces je navrhnut tak, aby spočítal počet chyb na konkrétním pracovním procesu. V tomto procesu budeme pracovat s tabulkou Defect.

6.2.1 Postup implementace procesu

Podobně jako u předchozího procesu je nutné vytvořit datový zdroj, protože každý balíček má své vlastní zdroje. Znovu je připojena databáze s názvem DWUHK_Anonymized. Dále je vytvořen nový balíček, ve kterém bude model vypracován. V panelu nástrojů je vytažen Data Flow Task. V záložce Data Flow je pomocí Source Assistant připojena tabulka Defect. Dalším krokem je vybrán v panelu nástrojů funkce Aggregate a propojena se zdrojovou tabulkou. Ve funkci aggregate jsou vybrány atributy, se kterými chceme provádět potřebné operace. U atributu id je zvolena operace Count z důvodu zjištění celkové počtu chyb daného eventu a seřazena podle atributu failevent.



Obrázek 25: Ukázka výběru atributů pro agregaci. Zdroj (vlastní).

Následně je pomocí Destination Assistant vytvořena tabulka se dvěma atributy CountOfDefects a failevent. V záložce Mappings bylo namapováno id s CountOfDefects a failevent s failevent. Po proběhnutí procesu se vytvořila tabulka CountOfDefect, která obsahuje celkový počet chyb na daném eventu. Nakonec se tento balíček nahraje do SSIS katalogu na MS SQL Serveru.

Z této tabulky lze vytvořit analytické závěry. Nejvíce chyb, které nastanou při výrobě produktu se vyskytují na eventu E0030 s počtem 2140, E0026 s počtem 1542 a dále na E0029 s počtem 689. Tímto výpočtem jsme zjistili eventy s nejvyšší chybovostí, které je potřeba modifikovat či opravit za účelem snížení nákladů na případnou opravu produktu.

CountOfDefects	failevent
2	E0011
6	E0018
1542	E0026
182	E0028
689	E0029
4	E0008
2140	E0030
1	E0031
27	E0035
20	E0036
6	E0037

Obrázek 26: Pohled na data v tabulce CountofDefects. Zdroj (vlastní).

6.3 Návrh třetího ETL procesu

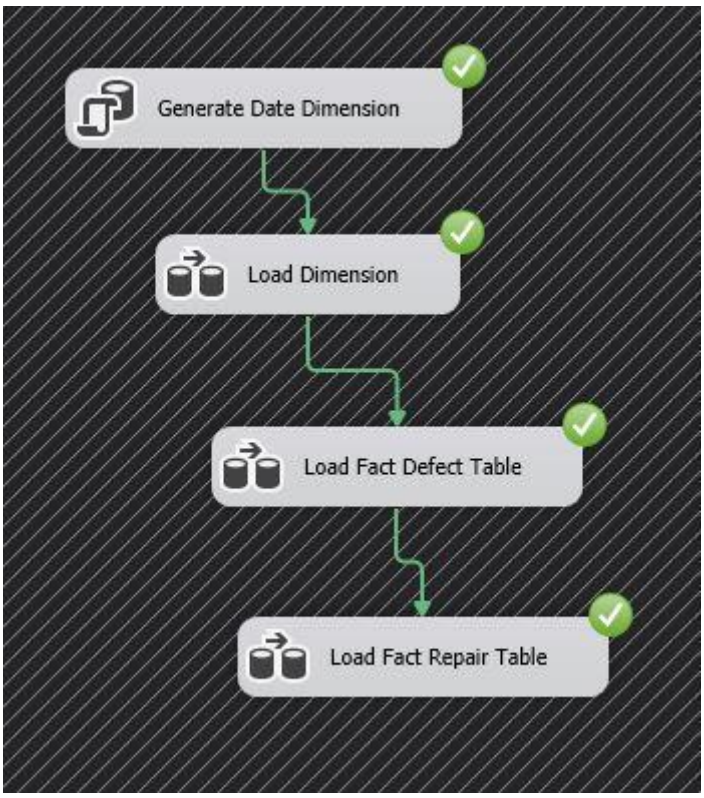
Třetí proces nahrává data do datového skladu. Tento proces využívá tabulky dimenzí a faktů, které byly navrženy při modelování datového skladu. Jako zdroj dat je použita tabulka Defect. Po úspěšném vykonání celého procesu se nahrají data do faktových tabulek.

faildescription	id_Time	id_Event	id_Product	id_TypeofDefect	id_User	id_Defect	id_Component
FC0400	5046187	2230	18255	4388	11	581	2322723
FC0178	5046197	2230	18255	4990	27	582	2322723
FC0176	5046204	2230	18255	4274	27	583	2322723
FC0176	5046204	2230	18255	4274	27	584	2322723
FC0239	5046191	2285	3241	356	27	585	1509950
FC0254	5046199	2608	20626	14	27	586	1643343
FC0239	5046186	2226	648	356	11	587	1509598
FC0378	5046191	2281	3313	215	35	588	1509901
FC0176	5046193	2311	1973	4274	27	589	1514183
FC0131	5046199	2785	12797	4391	27	590	1576094
FC0150	5046200	2785	12797	4255	11	591	1576094
FC0149	5046198	2590	1810	3970	27	592	1514792
FC0231	5046198	2605	39647	4382	27	593	2979154
FC0264	5046198	2605	39647	4281	11	594	2979154
FC0428	5046219	2605	39647	634	1	595	2979154
FC0149	5046198	2594	3223	3970	27	596	1515579
FC0149	5046200	2609	3090	3970	27	597	1515203
FC0149	5046198	2598	3009	3970	27	598	1513256
FC0239	5046198	2621	1764	356	27	599	1515758
FC0239	5046198	2621	1764	356	11	600	1515758
FC0239	5046203	2621	1764	356	11	601	1515758
FC0149	5046198	2603	1168	3970	27	602	1515398
FC0254	5046199	2603	1168	14	11	603	1515398
FC0254	5046199	2603	1168	14	27	604	1515398
FC0149	5046214	3299	103831	3970	27	605	893699
FC0134	5046206	3301	94379	3966	27	606	290056
FC0330	5046206	3396	3093	27	11	607	1522952
FC0330	5046206	3398	22048	27	11	608	2455194

Obrázek 27: Data faktové tabulky Defect. Zdroj (vlastní).

6.3.1 Postup implementace procesu

Tento proces je rozdělen na čtyři části. Jednotlivé části procesu se vykonávají postupně. Při vyskytnutí chyby v jedné části způsobí selhání celého procesu.



Obrázek 28: Náhled na celý ETL proces. Zdroj (vlastní).

Generate Date Dimension

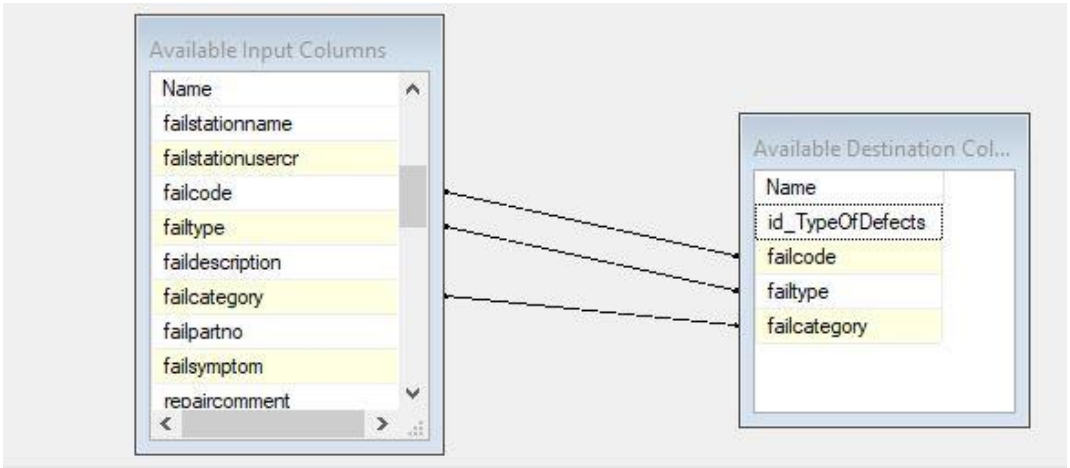
V první části procesu běží script, který vygeneruje data datumu v rozpětí od začátku roku 2017 do konce roku 2035. Každý řádek tabulky představuje jeden den v daném roce. Tyto data se uloží do dimenze Time.

	id_Time	datetime	year	month	day
1	5045817	2017-01-01 00:00:00.000	2017	1	1
2	5045818	2017-01-02 00:00:00.000	2017	1	2
3	5045819	2017-01-03 00:00:00.000	2017	1	3
4	5045820	2017-01-04 00:00:00.000	2017	1	4
5	5045821	2017-01-05 00:00:00.000	2017	1	5
6	5045822	2017-01-06 00:00:00.000	2017	1	6
7	5045823	2017-01-07 00:00:00.000	2017	1	7
8	5045824	2017-01-08 00:00:00.000	2017	1	8
9	5045825	2017-01-09 00:00:00.000	2017	1	9
10	5045826	2017-01-10 00:00:00.000	2017	1	10
11	5045827	2017-01-11 00:00:00.000	2017	1	11
12	5045828	2017-01-12 00:00:00.000	2017	1	12
13	5045829	2017-01-13 00:00:00.000	2017	1	13
14	5045830	2017-01-14 00:00:00.000	2017	1	14
15	5045831	2017-01-15 00:00:00.000	2017	1	15

Obrázek 29: Pohled na data dimenze Time. Zdroj (vlastní).

Load Dimension

V této části se nahrávají data do jednotlivých dimenzí. Zdroj těchto dat je tabulka Defect. Pomocí funkce Balanced Data Distributor rozdělíme jednotlivé atributy tabulky Defect mezi dimenze. V každé dimenzi v záložce mappings namapujeme atributy tabulky Defect, které chceme převádět.



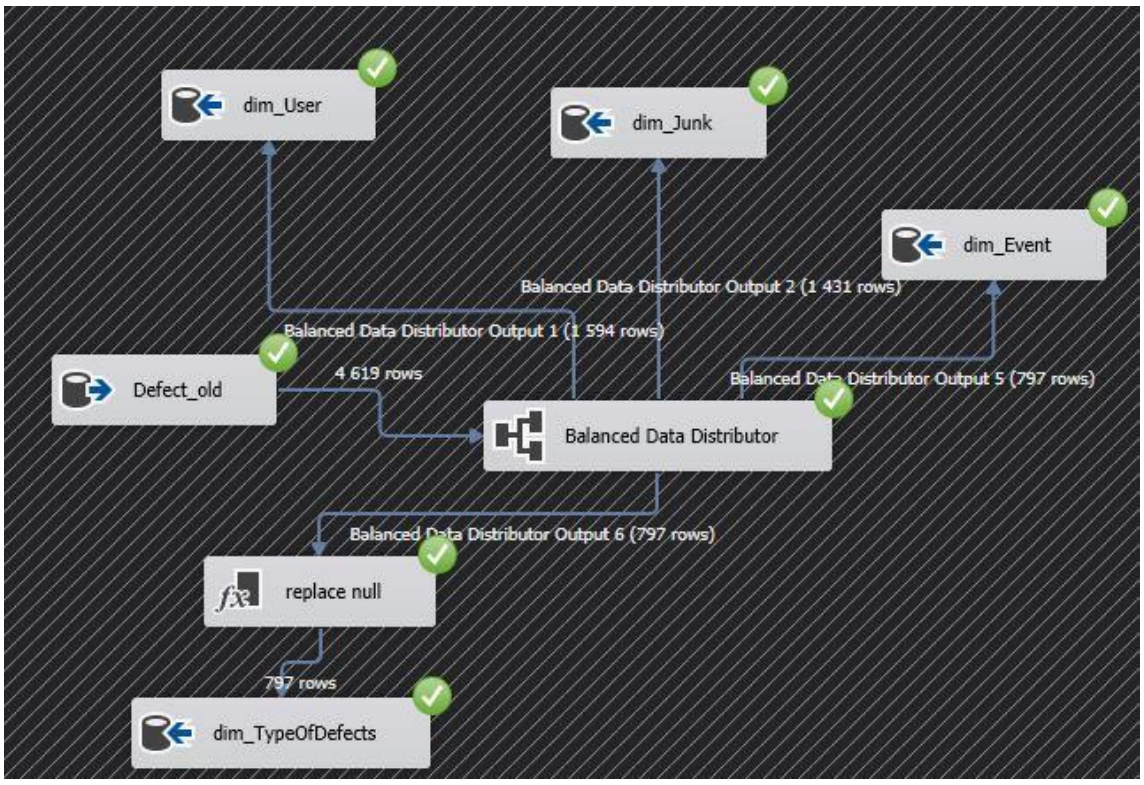
Input Column	Destination Column
<ignore>	id_TypeOfDefects
failcode	failcode
failtype	failtype
failcategory	failcategory

Obrázek 30: Mapování atributů TypeOfDefect dimenze. Zdroj (vlastní).

Dále je zde použita transformace nahrazení null hodnot před nahráním dat do dimenze TypeOfDefects. Pro vykonání této transformace slouží funkce Derived Column, která nahradí null hodnoty libovolnou hodnotou. Null hodnoty jsou nahrazeny slovem neuvedeno, abychom se vyhnuli null hodnotám v dimenzi.

Derived Column Name	Derived Column	Expression
failcode	Replace 'failcode'	REPLACENULL(failcode,"neuvedeno")
failtype	Replace 'failtype'	REPLACENULL(failtype,"neuvedeno")
failcategory	Replace 'failcategory'	REPLACENULL(failcategory,"neuvedeno")

Obrázek 31: Nahrazení null hodnot. Zdroj (vlastní).



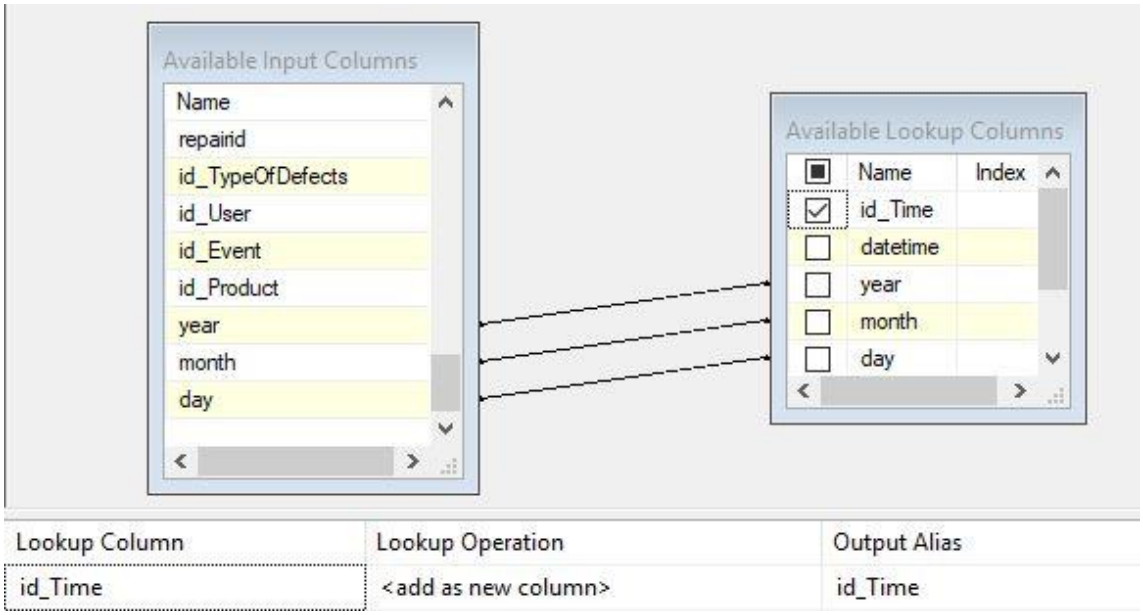
Obrázek 32: Pohled na celou část Load Dimension. Zdroj (vlastní).

Load Fact Defect Table

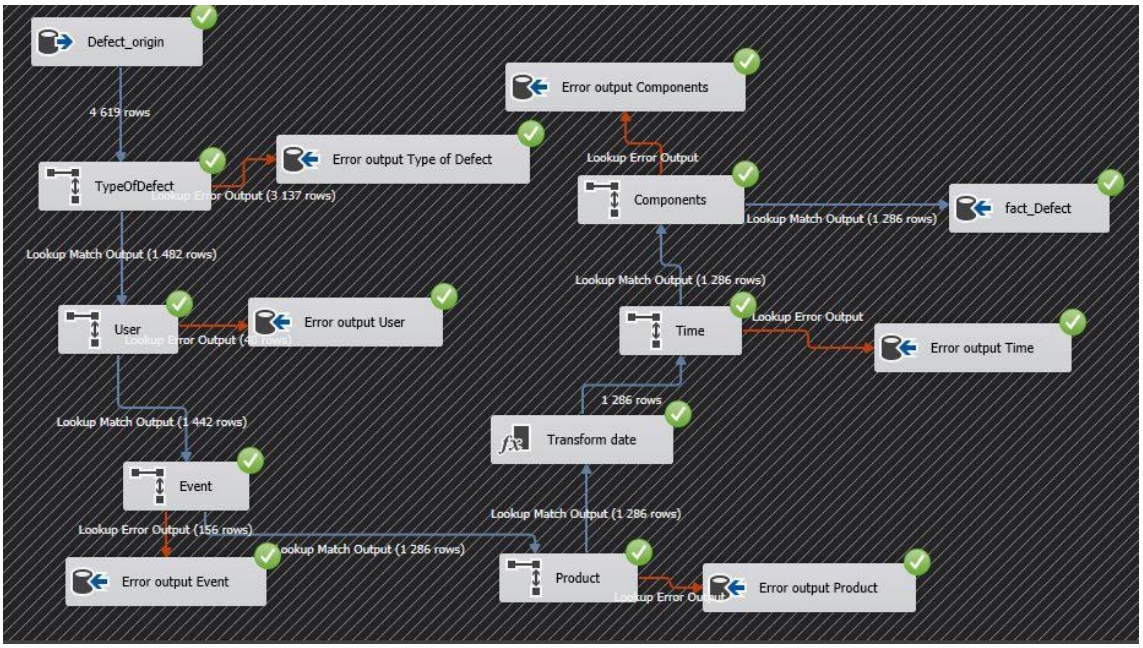
V této části probíhá nahrávání dat do faktové tabulky Defect. Zdroj dat je zde původní tabulka Defect. Jednotlivé dimenzionální tabulky jsou připojeny pomocí LookUp transformace. V každé dimenzi jsou namapovány atributy, které odpovídají atributům zdrojové tabulky Defect. Dále jsou u každé dimenze ošetřeny chyby, které mohou nastat při mapování a nahrávání do faktové tabulky. Jednotlivé chybné řádky se odklání do samostatné tabulky, z důvodu předejití selhání celého procesu. Před připojením dimenze Time, je datum transformován pomocí funkce Deliver Column. Tato transformace je potřeba pro následné připojení dimenze Time. Transformován je atribut faildate, který označuje datum nastání chyby. Transformací získáme z atributu faildate tři nové atributy, které se následně namapují s dimenzí Time.

Derived Column Name	Derived Column	Expression
year	<add as new column>	YEAR(faildate)
month	<add as new column>	MONTH(faildate)
day	<add as new column>	DAY(faildate)

Obrázek 33: Transformace atributu faildate. Zdroj (vlastní).



Obrázek 34: Mapování atributů dimenze Time. Zdroj (vlastní).



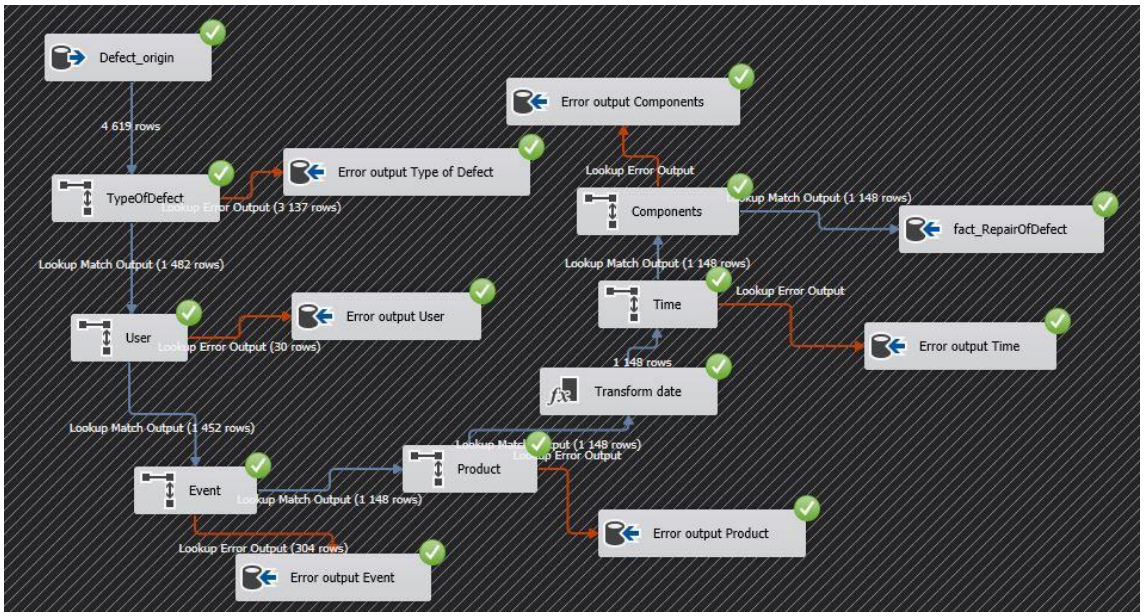
Obrázek 35: Pohled na celou část Load Fact Defect Table. Zdroj (vlastní).

Load Fact Repair Table

V poslední části jsou nahrávány data do faktové tabulky RepairofDefect. Dimenze jsou v tomto modelu sdílené tudíž je postup stejný jako v předchozí části. Jediná změna se týká mapovaných atributů některých dimenzí. V jednotlivých dimenzích jsou namapovány atributy tabulky Defect, které souvisí s opravou chyb. Dále je zde transformace datumu jako v předchozí části. Transformován je místo atributu faildate atribut repairdate. Atribut repairdate obsahuje datum opravy dané chyby. Transformací získáme tři atributy, které jsou ale rozdílné s předchozími. Získané atributy jsou následně namapovány v dimenzi Time.

Derived Column Name	Derived Column	Expression
year	<add as new column>	YEAR(repairdate)
month	<add as new column>	MONTH(repairdate)
day	<add as new column>	DAY(repairdate)

Obrázek 36: Transformace atributu repairdate. Zdroj (vlastní).



Obrázek 37: Pohled na celou část Load Fact Repair Table. Zdroj (vlastní).

7 Závěr

Cílem práce bylo analyzovat různé přístupy k řešení datového skladu a porovnat přístupy ke zpracování dat. V první části byly vysvětleny databázové systémy a jejich rozdělení. Byl pospán multidimenzionální model, jeho důležité části a typy. V další kapitole je detailně popsán datový sklad. Nejdříve jsou vysvětleny základní vlastnosti datového skladu. Dále je rozebrán postup vytváření datového skladu a popsány jednotlivé architektury. Následující kapitola se zabývá procesem ETL. V této kapitole je ETL proces detailně vysvětlen a popsán. Dále jsou popsány jednotlivé nástroje pro práci s ETL procesy.

Cílem praktické části bylo zanalyzovat datový model datového skladu a navrhnout ETL proces pro spolupracující společnost. Praktická část je tedy rozdělena na dvě části. V první části je popsán databázový model datového skladu společnosti a následně zanalyzován. Ve druhé části byly navrhnuty ETL procesy. Následně byl detailně popsán postup implementace jednotlivých procesů v prostředí MS SSIS.

8 Zdroje

[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] CEDR. Multidimenzionální kostka. In: *Cedr* [online]. [cit. 2019-01-26]. Dostupné z: http://cedr.mfcr.cz/Cedr3INetHelp/CommonPages/Terminologie0001.aspx

[3] ŽIŽKA, Jan. *Business intelligence*. Praha: Vysoká škola ekonomie a managementu, 2011. ISBN 978-80-86730-79-0.

[4] LABERGE, Robert. *Datové sklady, agilní metody a business intelligence*. Brno: Computer Press, 2012. ISBN 978-80-251-3729-1.

[5] LACKO, Ľuboslav. *Databáze: datové sklady, OLAP a dolování dat s příklady v Microsoft SQL Serveru a Oracle*. Brno: Computer Press, 2003. ISBN 80-7226-969-0.

[6] DATAWAREHOUSE4U. Datový sklad. In: *Datawarehouse4u.info* [online]. c2008-2019 [cit. 2019-01-22]. Dostupné z: https://www.datawarehouse4u.info/index_en.html

[7] INMON, William H. *Building the data warehouse*. 4th ed. Indianapolis, Ind.: Wiley, c2005. ISBN 0764599445.

[8] INMON, William H. *Building the data warehouse*. 3rd ed. New York: J. Wiley, c2002, 412 s. ISBN 04-710-8130-2

[9] KIMBALL, Ralph a Joe CASERTA. *The data warehouse ETL toolkit: practical techniques for extracting, cleaning, conforming, and delivering data*. Indianapolis, IN: Wiley, c2004. ISBN 0764567578.

[10] APPLIED INFORMATICS. ETL proces. In: *Applied Informatics* [online]. 2015 [cit. 2019-02-11]. Dostupné z: http://blog.appliedinformaticsinc.com/etl-extract-transform-and-load-process-concept/

[11] ADASTRA. ETL/ELT. *Adastra* [online]. 2016 [cit. 2019-04-06]. Dostupné z: https://www.adastra.cz/technologie/etl-elt

[12] POWERCENTER. In: *Capterra* [online]. [cit. 2019-04-06]. Dostupné z: https://cdn0.capterra-static.com/screenshots/530/149081.png

[13] ORACLE. Oracle Data Integrator. In: *Oracle* [online]. [cit. 2019-04-06]. Dostupné z: https://www.oracle.com/us/assets/im08t0-odi-ee-1-2034208.jpg

9 Seznam obrázků

10 Přílohy

Příloha 1: CD s praktickou částí

Příloha 2: Zadání práce