Univerzita Hradec Králové
Fakulta informatiky a managementu
Katedra informatiky a kvantitativních metod

Analýza a optimalizace procesu zpracování otevřených dat Královehradeckého kraje

Diplomová práce

Autor: Martin Gold
Studijní obor: DV-2

Vedoucí práce: Petra Poulová doc. RNDr. Ph.D.

Hradec Králové měsíc rok

Prohlášení:

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

vlastnoruční podpis

V Hradci Králové dne 26.4.2023 Jméno a Příjmení

Poděkování:
Děkuji vedoucí diplomové práce Petře Poulové doc. RNDr. Ph.D. za metodické vedení práce a Zbyňku Hálovi za odborné konzultace ohledně praktické části diplomové práce.

Anotace

Diplomová práce se zaměřuje na analýzu procesu zpracování dat v datovém portále Královéhradeckého kraje a návrh automatizace procesu s využitím datového skladu v prostředí Data KHK. Práce se zabývá jak jednoduchou automatizací z pohledu nahrazení lidské činosti tak i návrhem začlenění datového skladu do procesu zpracování dat.

Annotation

Title: Analysis and optimization of the data mining process of the Hradec Králové Region

The thesis focuses on the analysis of the data processing process in the data portal of the Hradec Králové Region and includes propose of process automation using the data warehouse in the Data KHK environment. The thesis deals with both simple automation in terms of replacing human activity and the proposal of incorporating the data warehouse into the data processing process.

Obsah

Seznam obrázků

Seznam tabulek

1 Úvod

Datový portál Královéhradeckého kraje slouží občanům k rychlému, jednoduchému a pohodlnému vhledu do otevřených dat kraje od dopravy přes školství po dotace. Portál má formu webové aplikace, kde jsou dostupné přehledné infografiky z různých odvětví činnosti kraje. Tato data pocházejí z mnoha zdrojů (např. Ministerstvo práce a sociálních věcí, Ministerstvo financí nebo kraj samotný). Tyto data pocházejí z mnoha zdrojů a nemají jednotný formát, protože je zveřejňují různé instituce a neexistuje standardizovaný formát pro výměnu těchto dat, proto neexistuje univerzální řešení a je třeba aplikovat techniky data miningu pro získávání, zpracování, ukládání a zobrazování těch dat.

Stávající řešení není zcela automatizované a v několika krocích je třeba ručního zásahu, při kterém mohou vznikat chyby a tyto zásahy jsou opakujícího se charakteru a vyžadují čas člověka. Tyto úlohy je možno automatizovat pro každý datový soubor pomocí technik data science popsaných v první části práce. Ve druhé části práce je zanalyzováno stávající řešení zpracování dat a je navržena implementace řešení pro vyšší stupeň automatizace a implementace datového skladu.

2 Cíl práce

Cílem této práce je analyzovat stávající řešení zpracování dat Královéhradeckého kraje a navrhnout automatizované řešení spočívající v návrhu datového skladu včetně celého řetězce od zdrojových dat až po výsledná data, která mohou být vizualizována. Pro účely proveditelnosti byl zvolen přehled vyočkovaných dávek vakcíny proti nákaze COVID-19.

Tento návrh datového sklad by měl sloužit jako centrální úložiště dat datového týmu KHK, tak aby data mohla být jednoduše aktualizována a jednoduše dostupná při zpracovávání budoucích vizualizací.

3 Kapitola - Vlastní text práce

3.1 Historie datové vědy

Datová věda není nijak novou disciplínou, vzniká již od sedmdesátých let dvacátého století. Až přibližně na přelomu tisíciletí se společnosti začali zajímat a vidět potenciál nashromážděných dat. Tato data byla složité získávat a až s velkým rozmachem internetu a mobilních zařízení jejich množství vrostlo. S tím vznikla potřeba toto nepřeberné množství dat nějakým způsobem analyzovat, vyhodnocovat a vizualizovat. (1)

Na obrázku č. 1 lze vidět postupný nárůst zájmu o termín „data science“ z internetového vyhledávače google.com mezi lety 2004 až 2023.

Obrázek 1 - Popularita termínu Data science
Zdroj: trends.google.com

3.2 Vymezení pojmů

3.2.1 Data vs. Informace

Běžně se tyto dva pojmy používají ve stejné kontextu a významu ovšem je třeba mezi nimi rozlišovat pro další použití a pochopení. Obecná definice říká, že data jsou souborem neuspořádaných faktů. Tyto informace jsou samotné hodnoty získané například dotazníkovým šetřením, měřením, senzoricky nebo jako výsledky ekonomické činnosti a jsou vytvářeny kontinuálně. Pokud se díváme na jednotlivá data o samotě, tak nedávají žádný větší obrázek. Samotná data není jednoduché jednoduše chápat, sdílet nebo z nich vyvozovat závěry. (1)

Informace jsou oproti tomu odvozené znalosti z dat, které nám poskytují větší obrázek o informacích a jsou již organizované a sestavené za nějakým účelem pro pozdější analýzu.

Například výška lidí jsou data. Ta reprezentují fakta reálného světa.

Žák	**Pohlaví**	**Známka**
Petr Novák	Muž	183
Kateřina Dostálová	Žena	178
Prokop Diviš	Muž	164

Tabulka 1 - Data o známkách žáků

Odvozením znalostí z těchto dat vzniká informace. Tou může být například, že průměrná naměřená výška žáků je 175. Tato informace je už pochopitelnější a dá se s ní lepe pracovat v kontextu porovnání např. mezi jednotlivými skupinami nebo pohlavím.

3.2.2 Dataset

Dataset je kolekce dat potřebných pro kompletní analýzu. Tabulka č.1 reprezentuje dataset o třech případech a třech proměnných. Jednotlivé sloupy jsou data. Jednotlivé řádky jsou případy (angl. „cases“). Ty reprezentují jednotlivá měření, objekty, skupinu nebo stav v čase. Dataset bývá nějčastěji reprezentován pomocí csv (comma separated values) soboru, ve kterém řádky představují případy a každý sloupec na řádku je oddělen čárkou. Pro jednoznačnou identifikaci může první řádek nést názvy proměnných též oddělené čárkou. (2)

3.2.3 Strukturovaná vs. Nestrukturovaná data

Strukturovaná data nabývají organizované podoby, kde každé pole má svůj jasně daný význam a místo. Dodržují předem danou strukturu, proto je možné jednoduše data analyzovat, modifikovat a přesouvat. V datech jsou jasně definované vztahy. Tyto data lze nalézt zejména v relačních databázích.

Semi-strukturovaná data jsou (často nazvána samo-popisná data) neposkytují takovou úroveň organizovanosti jak data strukturovaná. Nemají jasně definované vztahy. Oproti datům nestrukturovaným obsahují prvky, díky kterým lze vyextrahovat sémantiku dat od struktury. Nejčastějšími zástupci dat tohoto typu jsou data ve formátu CSV, JSON nebo XML.

Nestrukturovaná data, jak již název napovídá, neobsahují žádnou strukturu. Nedá se z nich jednoznačně extrahovat informace a pokud nějakou strukturu obsahují, tak se na ni nelze spolehnout. Díky pokročilým technikám Data Miningu se lze extrahovat informace z psaných textů a dalších nestrukturovaných zdrojů. (3)

3.2.4 Datové typy

Data samotná mohou být několika typů. Nelze například porovnávat například čísla bez znalosti jejich významu. Mezi základní datové typy patří:

Kvantitativní – též numerické, které značí obyčejné číselné hodnoty nejčastěji reprezentující nějakou fyzikální veličinu (teplota, tlak, délka, počet, …). Bývají nejčastěji zastoupena a práce s nimi bývá nejjednodušší, protože lze z nich jednoduše získat informaci například jednoduchým průměrem.

Nominální – též kategorické, které vypovídají o zařazení prvku do některé ze dvou či více skupin. Na příkladu z předchozí kapitoly se jedná o sloupec Pohlaví.

Ordinální – také vypovídají o zařazení prvku do kategorie, ovšem s rozdílem, že lze mezi sebou porovnávat a řadit. Obecně jde říct, že žák v osmé třídě je vyšší třídě než žák ve čtvrté, ovšem nejedná se o numerický údaj. (2)

3.2.5 Formáty dat

Formáty dat vznikly z potřeby standardizace, tak aby data mohla být ukládána a čtena různými stranami. Tyto formáty se značí koncovkou v názvu souboru za tečkou.

Mezi nejběžnější formáty patří CSV (comma separated value). Výhodou toho to formátu je jednoduchost, čitelnost člověkem i počítači. Každý řádek v souboru představuje jeden případ a data jsou na řádku oddělena čárkou. První řádek může obsahovat záhlaví, které není případem, ale dává název každé proměnné ve sloupci. Nevýhodou tohoto formátu je absence možnosti reprezentace zanořených dat.

```
name,height,gender
"Petr Novák",183,"male"
"Kateřina Dostálová", 178,"female"
"Prokop Diviš",164,"male"
```

Nejběžnějším formátem pro výměnu dat ve webovém prostředí je JSON (JavaScript object notation). Oproti CSV lze zapisovat i vnořené vazby, ovšem formát je již složitější a není úplně triviální oddělit sémantiku dat jako u CSV. Existují ovšem knihovny v téměř všech programovacích jazycích, které prací s formátem JSON umožňují. Podporuje datové typy jako je řetězec, desetinné číslo, pravdivostní hodnota, objekt, pole nebo speciální null hodnota.

```
[
    {
        "name": "Petr Novák",
        "height": 183,
        "gender": "male"
    },
    {
        "name": "Prokop Diviš",
        "height": 164,
        "gender": "male"
    }
]
```

XML (Extensible Markup Language) je značkovací jazyk pro serializaci dat do textové podoby. Umožnuje i validace pomocí formátu XSD. Obecně ze všech tří formátu nabízí nevětší možnosti pro zápis dat.

```
<?xml version="1.0" encoding="UTF-8" ?>
<people>
    <person>
        <name>Petr Novák</name>
        <height>183</height>
        <gender>male</gender>
    </person>
    <person>
        <name>Kateřina Dostálová</name>
        <height>178</height>
        <gender>female</gender>
    </person>
    <person>
        <name>Prokop Diviš</name>
        <height>164</height>
        <gender>male</gender>
    </person>
</people>
```

3.3 Fáze na projektu

Projekt datové analýzy je třeba rozdělit do několika fází kde každá fáze má své předpoklady a cíle. Pro každou fázi je třeba rozdílných rolí, kdy například při vymezování cílů není nezbytně nutné, aby se jí účastnil datový inženýr.

3.3.1 Vymezení cíle datové analýzy

Před zahájením jakýchkoliv prací je nutné se zamyslet jaké jsou cíle a stanovit si základní otázky typu:

- Čeho chceme dosáhnout?
- Na základě čeho budeme měřit změny?
- Jsme schopni sehnat data pro dosažení cíle?
- Co považujeme úspěch/neúspěch?
- Jak můžeme aplikovat výsledky?
- Jaké jsou primární metriky?

Vymezení těchto cílů na fiktivním fitness centru může vypadat následovně:

Datovou analýzou chceme dosáhnout vyšší návštěvnosti a vyšší vytíženosti fitness centra. Hlavní metrikou pro měření úspěchu je zisk a spokojenost návštěvníků. Tato data můžeme obstarávat prostřednictvím emailových dotazníků a z pokladního systému, ve kterém se eviduje každá návštěva fitness centra. Za úspěch je považováno meziroční zvýšení zisku o 8 %. Můžeme investovat do lepšího vybavení, změnit výši vstupného a nastavovat akční slevy na dlouhodobé vstupné.

3.3.2 Získávání dat a čištění dat

Pro analýzu je třeba zajistit data samotná. Ty mohou pocházet z různých zdrojů. Některá data lze čerpat z již existujících informačních systémů napojením na API nebo z exportů v semi-strukturovaných formátech, které aplikace poskytuje. Dalším zdrojem mohou být dotazníková šetření, data naměřená ručně nebo získána senzoricky.

Ať jsou data získána jakkoliv, tak je třeba provést čištění dat, které spočívá v kontrole maximálních a minimálních hodnotách (např. věk 180 let bude s největší pravděpodobností neplatná hodnota), kontrola chybějících hodnot a v případě manuálního vstupu dat kontrola integrity jednotlivých datových typů (číselná pole v dotazníku obsahující znaky apod.). Je třeba znát doménu dat a kontrolovat, zda jsou přípustná a případně tato data vyřadit. (3)

3.3.3 Transformace dat

Jakmile jsou data pročištěna, tak třeba je seřadit a připravit do podoby vhodné pro analýzu. Pokud je k analýze potřeba více zdrojů, tak je vhodné je spojit v této fázi. Je zde i vhodné vybrat pouze relevantní data potřebná pro analýzu (např. pro analýzu realitního trhu v Hradci Králové není třeba mít data ostatních krajů) (3)

3.3.4 Analytické fáze

V této fázi se získává vhled do dat a souvislosti mezi nimi. Vytvářejí se nové odvozené informace. Získávají se klíčové metriky k zodpovězení otázek v první

fázi. Tyto informace jsou agregací zdrojových dat. Na příkladu fitness centra by zisk jako klíčová metrika byla vypočtena následovně:

$$z = \sum p - \sum v$$

Kde p jsou přijmy, v výdaje a z je celkový zisk. Těchto metrik může být několik.

3.3.5 Prezentace a aplikace výsledků

Tyto výsledky je třeba interpretovat a vyvodit rozhodnutí pro zodpovězení otázek položených v první fázi. Po ustanovení těchto rozhodnutí je třeba je aplikovat do praxe a pozorovat změny. Záleží na konkrétní implementaci, ovšem je důležité, aby bylo možné analýzu znovu spustit s novými daty, tak aby bylo možné pozorovat vliv rozhodnutí. Případně je třeba upravit rozhodnutí, aby bylo možné dosáhnout kýženého cíle.

Na příkladu fitness centra by z analýzy mohlo vyplynout, že v určitou denní dobu vzniká špička a návštěvnost centra je příliš velká a lidé nejsou spokojení, zatímco v jiné denní doby je centrum prázdné. Proto vznikne rozhodnutí zavést rozdílný ceník vstupného pro různé denní doby. Po nějaké době je třeba analýzu provést znovu, aby se zjistilo, jaký vliv měla daná rozhodnutí vliv na metriky a případně je třeba vydat rozhodnutí jiné.

3.4 Role v datovém týmu

Role datový vědec bývá často používána jako souhrnná role všech rolí v oboru. Datová věda je rozsáhlým oborem, kde není možné, aby jeden člověk obsáhnul všechny odpovědnosti. Proto se obor děli do více rolí (dle týmu Berkley Extension (5)), které dohromady tvoří datový tým.

3.4.1 Datový analytik

Datový analytik bývá vstupní pozice do oboru datové vědy. Primárním cílem této role je vyvíjet systémy pro sběr dat, třídit data organizace a vytvářet jednoduché vhledy do dat, které zodpoví na základní otázky. Lidé zastávající tuto pozici by

měli být všímavý, otevření diskusím a schopni analytického myšlení. Velmi důležitou schopností je komunikace, jelikož lidé na této roli často komunikují napříč odděleními a s lidmi zastávající různé role.

3.4.2 Datový vědec

Datový vědec zastává podobné zodpovědnosti jako datový analytik, ovšem jeho hlavním cílem je používat pokročilejší techniky, algoritmy a sestavovat modely. Datový vědec rozumí odkud data pocházejí a jak jsou zpracovány, ovšem již nepracuje se surovými daty. Mezi jejich hlavní přednosti patří především samostatnost, znalost pokročilých metod a zvídavost, jelikož jejich role je oproti ostatním více nezávislá.

3.4.3 Business analytik

Business analytik se nezabývá technickou stránkou datové analýzy, ovšem komunikuje s jednotlivými členy týmu, tak aby jim bylo jasné, co je cílem datové analýzy. Jeho další zodpovědností je převést potencionální výsledky analýzy na business rozhodnutí, která se dají aplikovat a komunikovat je se zbytkem organizace. Může nabývat i dalších zodpovědností, jako například kontrola kvality, time management nebo projektové řízení. Mezi jeho přednosti musí patřit pochopení procesů, kvalitní komunikační schopnosti a leadership.

3.5 Metodologie

Principy těžení dat a znalostí mají určité společné fáze, které se dají různě popsat, proto vznikly následující metodiky za cílem usnadnit práci s daty.

3.5.1 CRISP-DM Framework

CRISP-DM (The Cross Industry Standard Process for Data Mining) je rozsáhlá data-miningová meteologie a procesní model, který poskytuje všem, od nováčků po experty, kompletní šablonu pro vedení data-miningového projektu. CRISP-DM rozpadá celý projekt do šesti fází, a to porozumění na úrovní businessu, porozumění datům, příprava dat, modelování, ohodnocení a uvedení do praxe. (6)

Obrázek 2 – Životní cyklus fází v metodologii CRISP-DM
Zdroj: https://www.butleranalytics.com/crisp-dm-in-a-nutshell/

3.5.1.1 Porozumění businessu

Pravděpodobně nejdůležitější fází jakéhokoliv data-mining projektů je počáteční porozumění na business úrovní, která se soustředí na porozumění projektu z business hlediska a převedení těchto znalostí na definici data-mining otázky. Poté následuje vyvinutí předběžného plánu navrženého pro dosažení cílů. Aby bylo možné zjistit která data a jak je analyzovat, je důležité naprosto rozumět odkud se data berou a co znamenají na úrovni businessu. Tato fáze obsahuje několik kroků.

3.5.1.2 Porozumění datům

Další fází této metody je porozumění datům. Tato fáze začíná prvotním sběrem dat. Poté analytik dále data zkoumá a zevrubně se s nimi seznamuje, tak aby dokázal identifikovat problémy s kvalitou dat, získat jednoduchý vhled do dat, dokázal vytipovat důležité podmnožiny dat.

3.5.1.3 Příprava dat

Třetí fází je příprava dat. To zahrnuje veškerou práci s daty od surových dat až po data ve formátu, ve kterém jsou použita pro sestavování modelu. Sestává se z několika kroků, například výběru dat, čištění dat od neplatných hodnot, konstrukce dat (odvození nových dat z již existujících), integrace dat (spojení dvou či více zdrojů do jednoho) a formátování dat.

3.5.1.4 Modelování

Fáze modelování se zabývá sestavením modelu, který data nejlépe reprezentuje. Obvykle existuje více technik a je třeba vybrat tu správnou. Některé techniky vyžadují data ve specifickém formátu, proto je občas nutné se vrátit do fáze přípravy dat a data připravit ve formátu která je vyžadován vybranou technikou. Tato fáze má několik kroků, a to výběr správné techniky, navržení testu pro ověření správnosti modelu, vytvoření a zhodnocení modelu.

3.5.1.5 Hodnocení modelu

Předposlední fází je hodnocení modelu, zda správně modeluje data. Zde je třeba důsledně zkontrolovat, zda je model schopen zodpovědět otázky položené v první fázi a zda se model není zaměřen pouze na některou z nich. Projektový vedoucí se musí rozhodnout, jak s výsledným modelem naložit. Důležitými kroky je zhodnocení výsledku, kontrola všech kroků vedoucích k modelu a určení další kroků.

3.5.1.6 Nasazení

Sestavení modelu nebývá poslední fází projektu, ovšem nasazení modelu. Získané znalosti musí být reorganizovány a prezentovány, tak aby jim koncový uživatel celého projektu dokázal porozumět a využít je. Tento model nenasazuje datový tým, ale samotný zákazník, který musí dopředu rozumět co nasazování takového modelu obnáší. Toho bývá dosaženo obsažením získaného modelu do rozhodovacího procesu organizace jako nástroje. Samotné nasazení se skládá

z vytvoření strategie nasazení, plánování a údržby, vytvoření finálního reportu a končeného ohodnocení projektu.

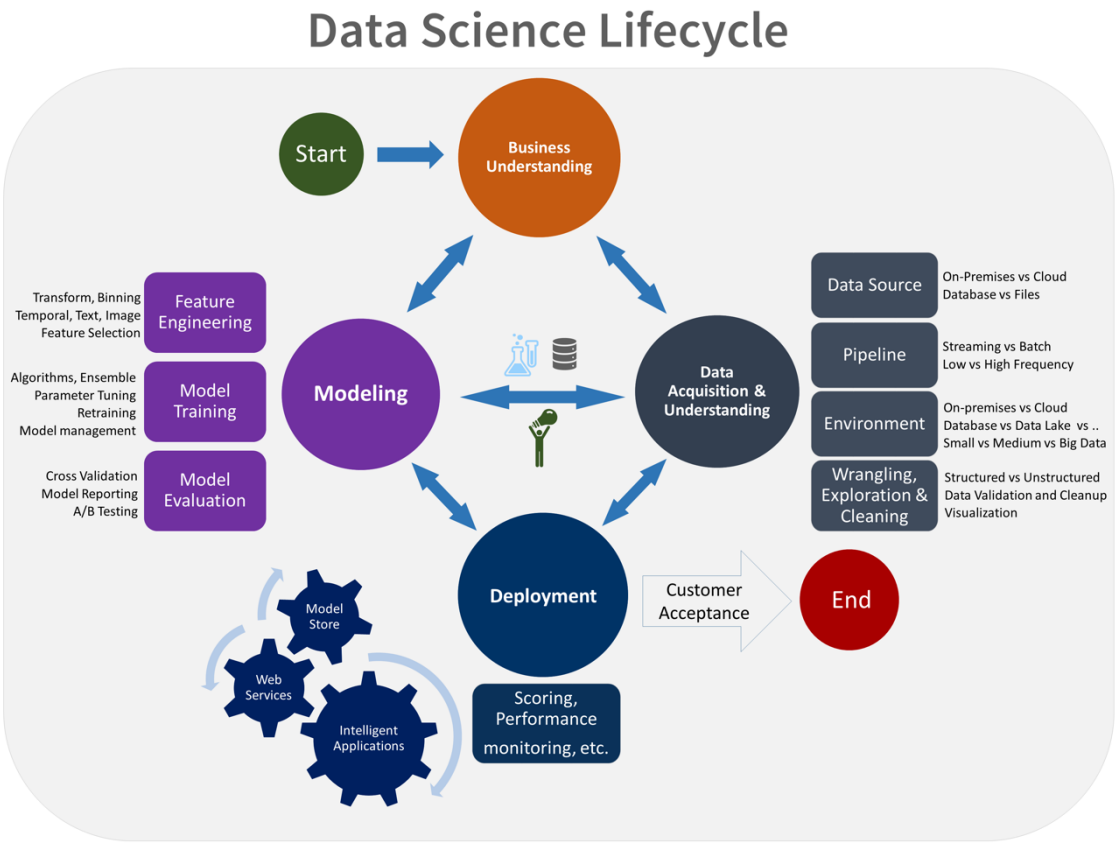
3.5.1.7 Shrnutí CRISP-DM

Celý CRISP-DM byl navržen, tak aby poskytoval pomocnou ruku pro začátečníky v oblasti data-miningu a poskytnul obecný procesní model, který jde přizpůsobit pro konkrétní odvětví nebo společnost. Neměl by sloužit jako příručka, nýbrž jako sada doporučení a šablona. Je založen na praktických zkušenostech z oblasti data-miningu. Vznikl ovšem v roce 1996 a dnes již není plně aktuální s nejnovějšími trendy.

3.5.2 Microsoft TDSP

Microsoft TDSP (Team Data Science Process) je agilní, iterativní metodologie datové vědy k řešení prediktivní analýzy a inteligentních aplikací. Pomáhá s týmovou spoluprací díky doporučením, jak jednotlivé role datového týmu mají spolupracovat. Cílem celé metodologie je pomoci společnostem a organizacím plně využít data nejen z pohledu analýzy, ale i z pohledu kontinuálního sběru dat, nasazení modelu po technické stránce, strukturou projektu a testováním (7)

Vznikl v roce 2016 a proto počítá s novějšími technologiemi jako je verzovací systém Git, Data lakes, nebo Big Data, které v době vzniku metodologie CRISP-DM neexistovaly.



Obrázek 3 - Diagram procesu TDSP
Zdroj: [7]

Tato metodika je svým rozsahem obsáhlejší než CRISP-DM. Neposkytuje pouze životní cyklus projektu. Skládá se z následujících komponent:

1. Životní cyklus datového projektu
2. Standardizovanou strukturu projektu
3. Doporučení pro infrastrukturu a zdroje projektu
4. Nástroje a pomůcky pro provedení projektu

3.5.2.1 Životní cyklus projektu

Životní cyklus projektu se od metodologie zásadně neliší od CRISP-DM. Fáze jsou následující:

1. Porozumění businessu
2. Získaní a porozumění datům
3. Modelování
4. Nasazení
5. Přijetí zákazníkem

V tomto se oproti CRISP-DM metodologii liší velmi málo, akorát fáze 2 a 3 je spojena dohromady a poslední fáze se zaměřuje spíše na dokončení projektu a jeho předání zákazníkovi.

3.5.2.2 Standardizovaná struktura projektu

Díky struktuře projektu vycházející ze šablony je pro všechny členy týmu jednodušší se v projektu orientovat a nalézt potřebné informace. Veškerý kód a dokumenty jsou ukládány pomocí verzovacího systému (Git, Mercurial, TFS), tak aby mohli všichni členové jednoduše spolupracovat. Řízení jednotlivých úloh je konáno pomocí systému pro agilní plánování projektu (Kanban, JIRA). Díky tomu lze lépe odhadnout náročnost celého projektu a jednodušeji sledovat postup.

3.5.2.3 Doporučení pro infrastrukturu a zdroje projektu

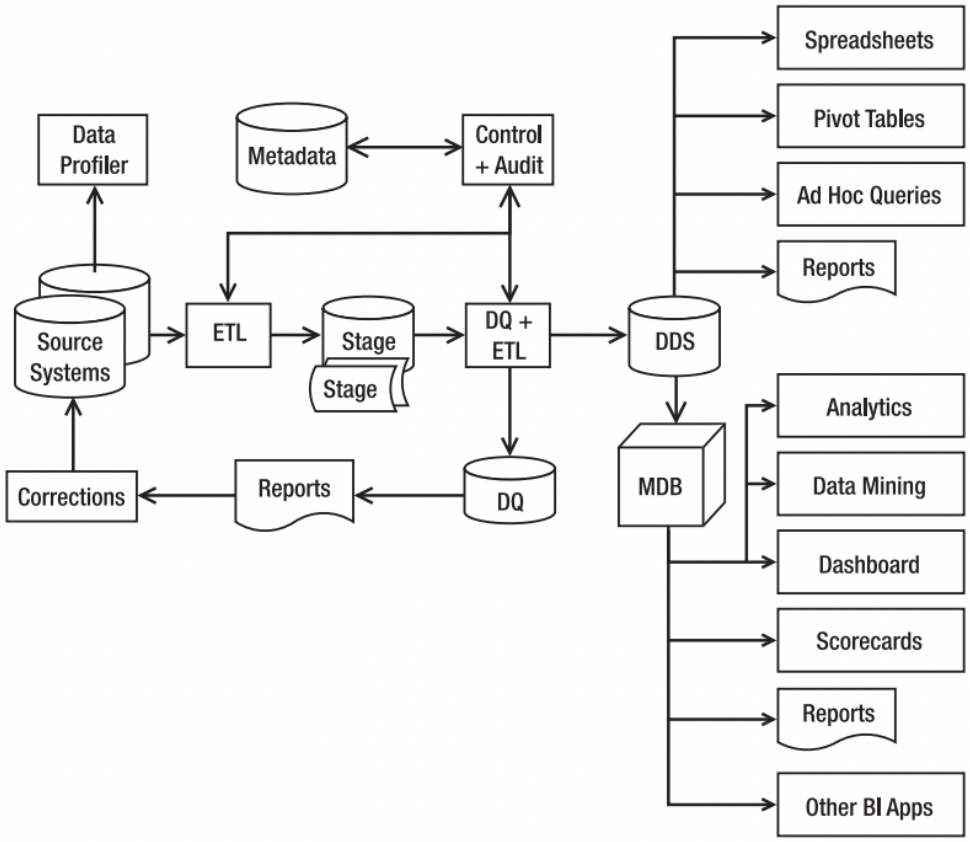
Metodika poskytuje doporučení pro správu celé infrastruktury, kde jsou data ukládána a kde jsou učeny modely. Tato infrastruktura se vyhýbá duplikaci dat, tudíž náklady na udržení a provoz jsou snižovány. Každý člen má sdílený přístup ke každému zdroji, tudíž je možná spolupráce.

3.5.2.4 Nástroje a pomůcky pro provedení projektu

TDSP poskytuje základní nástroje (explorativní analýza data a další) a mechanismus pro sdílení nástrojů napříč projekty, tak aby bylo možné začít používat metodiku co nejjednodušeji.

3.6 Datový sklad

Datový sklad je systém, který získává a konsoliduje data periodicky ze zdrojových systémů do dimenzionálního nebo normalizovaného datového úložiště. Obvykle obsahuje historii dat za několik let zpět a je dotazován za business nebo analytickými účely. Obvykle jsou data v datovém skladu aktualizována dávkově, ne pokaždé když se provede nějaká transakce. (8)



Obrázek 4 - Schéma datového skladu
Zdroj: Rainardi (8)

Na obrázku č.4 je zobrazen celý diagram databázového skladu. Zdrojové systémy (Source Systems) jsou OLTP (Online Transaction processing) systémy, jejichž úkolem je zachycovat a ukládat business transakce. Tyto data jsou dále vytahována a zpracována ETL/ELT procesem a ukládáno do DDS (Dimensional

Data Store). Tyto data v DDS jsou poté analyzovány a zkoumány uživateli pomocí různých nástrojů (SQL, reportovací nástroje, tabulkové procesory). Některé tyto nástroje (Dashboardy, BI nástroje) pracují s multidimenzionálními daty, proto jsou z DDS nahrány do MDB (Multidimensional Database).

3.6.1 OLTP

OLTP (Online Transaction Processing) jsou systémy které zpracovávájí klientské transakce jako jsou např. bankovní transakce, pokladní transakce nebo zasílání zpráv. Tyto transakce jsou obvykle nazývány jako finanční transakce, byť se vždy nemusí týkat financí. Tyto systémy mohou například sbírat senzorická data z turniketů, teploty z prostředí, zobrazení reklamy uživatelem nebo komentáře na sociálních sítích. Hlavními cíli těchto systémů je vysoká obslužnost mnoha klientů, bleskurychlá odezva, atomicita dat (nelze je měnit dvěma uživateli zároveň), vysoká spolehlivost a bezpečnost.

Data v těchto systémech bývají ukládána v ER (entity relation) modelu ve třetí normálové formě (3NF) v relačních databázích tak aby bylo docíleno co nejjednodušší modifikace informace pouze na jednom místě a odstranění redundance informací. (9)

3.6.1.1 1. Normálová forma

Tak aby byla splněna první normálová forma, tabulka nesmí obsahovat více sloupců s podobnou informací. Na příkladu si to lze představit tak, že místo toho abychom u tabulky objednávka přidávali sloupce pro každou položku nový sloupec „polozka1“, „polozka2“, „polozka3“, atd. tak je správné tyto položky.

zakaznik	adresa	objednavka1_cena	objednavka2_cena
Petr Novák	Na soutoku 1, Praha	334.31	213.7
Alena Dlouhá	Příkopy 23, Brno	1113.45	853.9

Tabulka 2 - Data nesplňující 1NF
Zdroj: Vlastní tvorba

Po normalizaci do 1NF by tabulka vypadala následovně:

zakaznik	adresa	objednavka_cena
Petr Novák	Na soutoku 1, Praha	334.31
Petr Novák	Na soutoku 1, Praha	213.7
Alena Dlouhá	Příkopy 23, Brno	1113.45
Alena Dlouhá	Příkopy 23, Brno	853.9

Tabulka 3 - Data splňující 1NF
Zdroj: Vlastní tvorba

3.6.1.2 2. Normálová forma

Druhá normálová forma říká, že tabulka obsahující duplicitní informace musí být rozdělena na více tabulek a spojena pomocí cizího klíče. Na předchozím příkladu by tabulka musela být rozdělena na dvě tabulky „objednavka“ a „zakaznik“ a poté spojena pomocí cizho klíče.

zakaznik_id	jmeno	adresa
1	Petr Novák	Na Soutoku 1, Praha
2	Alena Dlouhá	Příkopy 23, Brno

Tabulka 4 - Tabulka zákazníku v 2NF
Zdroj: Vlastní tvorba

zakaznik_id	objednavka_cena
1	334.31
1	213.7
2	1113.45
2	853.9

Tabulka 5 - Tabulka objednávek v 2NF
Zdroj: Vlastní tvorba

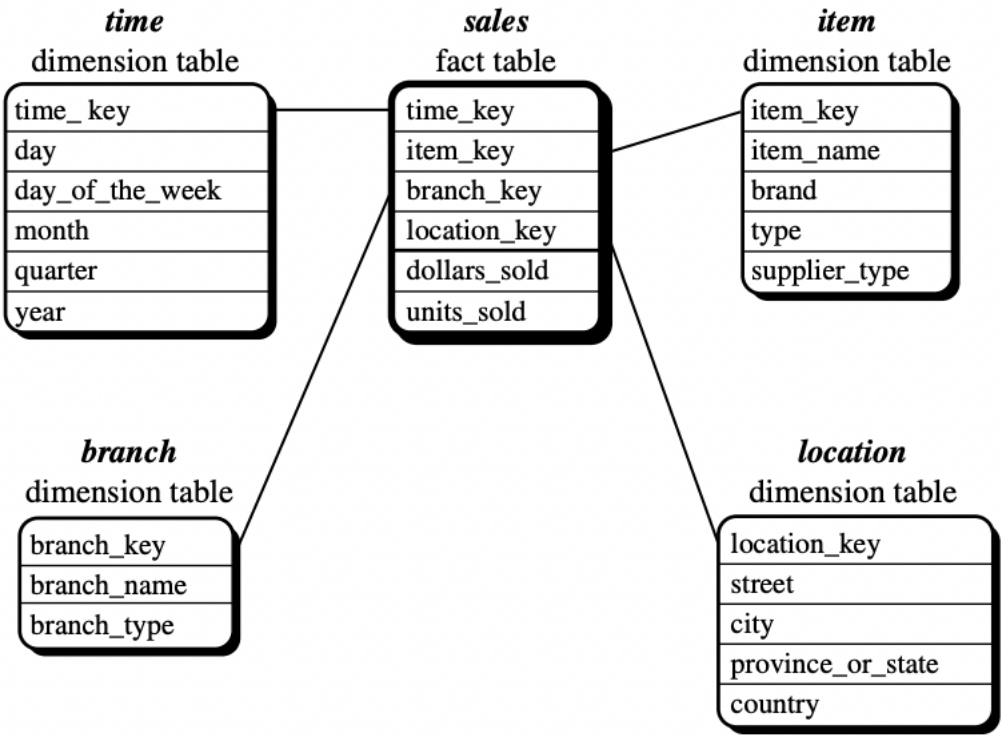
3.6.1.3 3. Normálová forma

Třetí normálová forma říká, že každý sloupec, který není funkčně závislý na primárním klíči musí být eliminován. Kdykoliv se hodnota sloupce může potencionálně týkat více záznamů, tak je třeba jej vyčlenit do zvláštní tabulky. Na předchozím případě by se tato forma dala uplatnit na adrese zákazníka (protože dva zákazníci mohou bydlet na stejné adrese) a potom by již byly splněny všechny tři normální formy. Ovšem to stejné se dá tvrdit i o jméně zákazníka (opět může být stejné) ovšem tabulka se jmény je nepraktická a poté již není z technických a praktických důvodů nutné normalizovat všechny sloupce.

3.6.2 OLAP

OLAP (Online Analytical Processing) je systém, který slouží především k multidimenzionální analýze klientských dat nasbíraných OLTP systémy. Tyto systémy jsou charakterizovány ohromným množstvím dat, proto jejich výkon není tak velký jako u OLTP systémů. Zároveň k nim přistupuje řádově nižší počet uživatelů a žádný z uživatelů neprovádí nad daty změny, pouze je čte. (9)

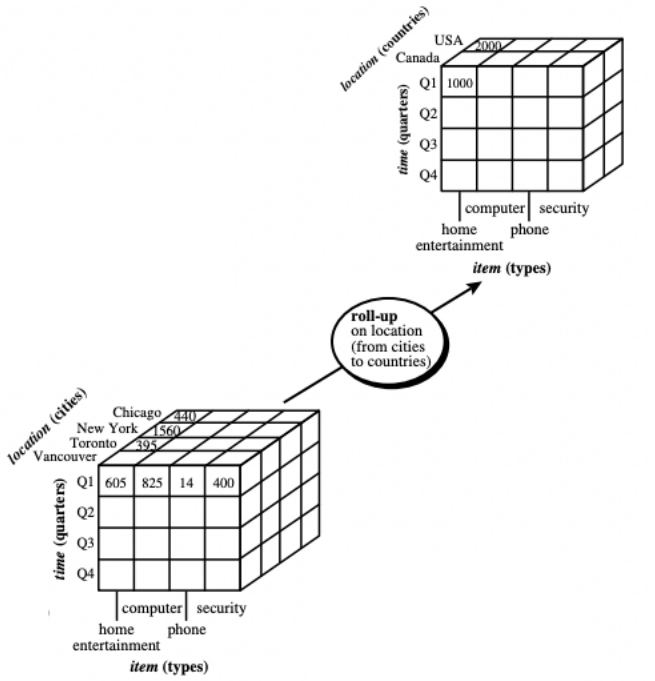
OLAP systémy využívají k ukládání dat hvězdicové schéma, které využívá jedné faktové tabulky (tabulka držící si primární klíče všech ostatních tabulek) a poté dalších subjektově zaměřených dimenzionálních tabulek (např. zákazník, objednávka, položka, adresa) kde každá tato tabulka má jeden cizí klíč na faktovou tabulku. Tímto je možné jednoduchými dotazy zjistit různé metriky.



Obrázek 5 - Příklad hvězdicového schématu
Zdroj: Han, Jiawei, Kamber

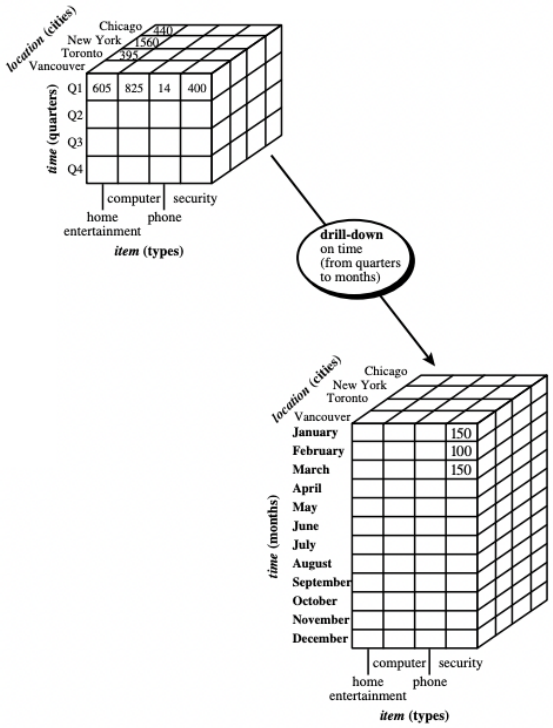
3.6.2.1 OLAP Operace

Rollup je operace při které se pohled dle kterého se data agregují zvýší o jednu úroveň (data nejsou zobrazena podle měst, ale podle států). Dalo by se říct, že se zvýší agregovanost dat od konkrétnějšího k obecnějšímu.



Obrázek 6 - Vizualizace operace Roll-up
Zdroj: Han, Jiawei, Kamber

Drill-down je operace opačná k roll-up, kdy se od pohled dostává od obecného ke konkrétnějšímu. Poskytuje detailnější pohled a je možné tak detekovat lépe anomálie, pokud se například zjistí nadprůměrné prodeje ve třetím kvartále roku, tak dril-down na třetí kvartál poskytne detailnější pohled toho, co se v daném kvartále dělo.

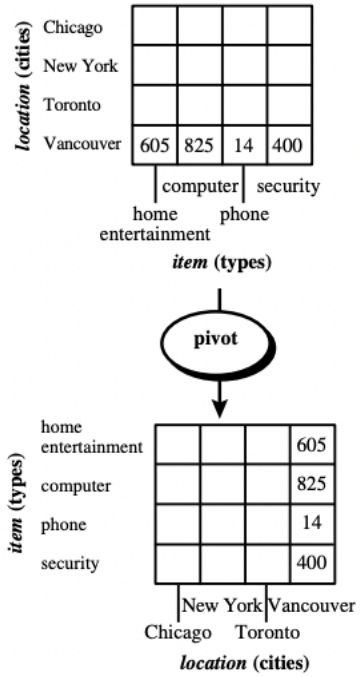


Obrázek 7 - Vizualizace operace Drill-down
Zdroj: Han, Jiawei, Kamber

Slice je operace, při které se vybere pouze jedna dimenze z OLAP kostky a tím vznikne tabulka. Pokud bychom provevedli operaci slice na OLAP kostce z obrázku č.6 dle nějakého časového období (1. kvartál nebo Říjen), tak bychom dostaly dvourozměrnou tabulku.

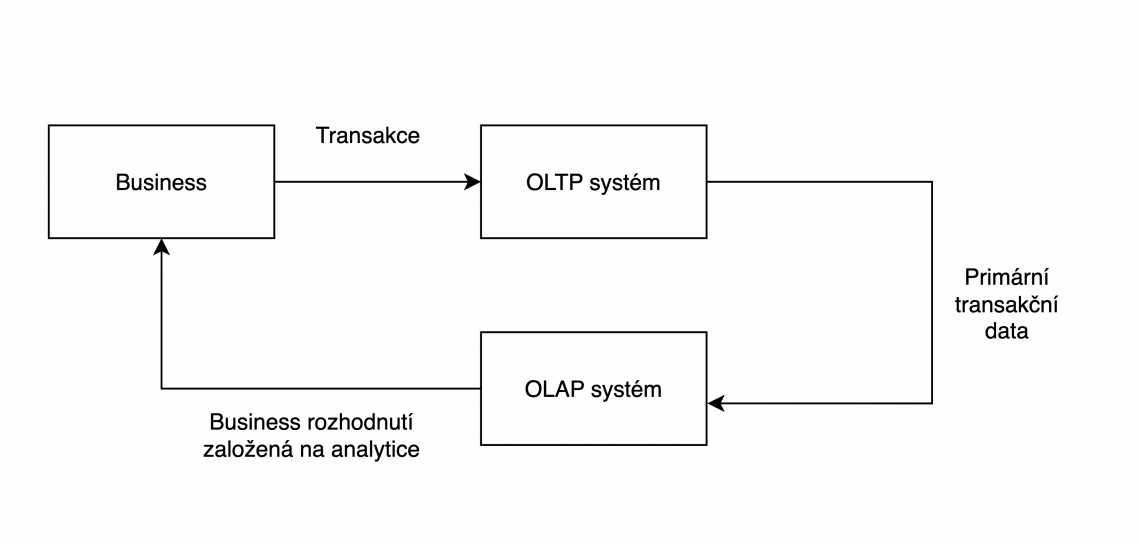
Dice je operace při které vybereme z každé dimenze pouze několik údajů. Pokud bychom provedli dice na OLAP kostce z obrázku č. 6 dle prvních dvou kvartálů a dle měst Vancouver a Toronto, tak bychom dostali pohled na prodeje v druhé polovině roku v Kanadě.

Pivot je oprace při které prohodíme osy a tím získáme kompletně jiný náhled na data.



Obrázek 8 - Vizualizace operace Pivot
Zdroj: Han, Jiawei, Kamber

3.6.3 Vztah OLTP a OLAP



Obrázek 9 - Vztah OLTP a OLAP systémů
Zdroj: Vlastní tvorba

Z předchozího diagramu by se mohlo zdát, že OLAP systém jsou zbytečné a analýzu by mělo být možné provádět rovnou na OLTP systému a tím si ušetřit složitost a náklady na implementaci celého datového skladu. Ovšem OLAP

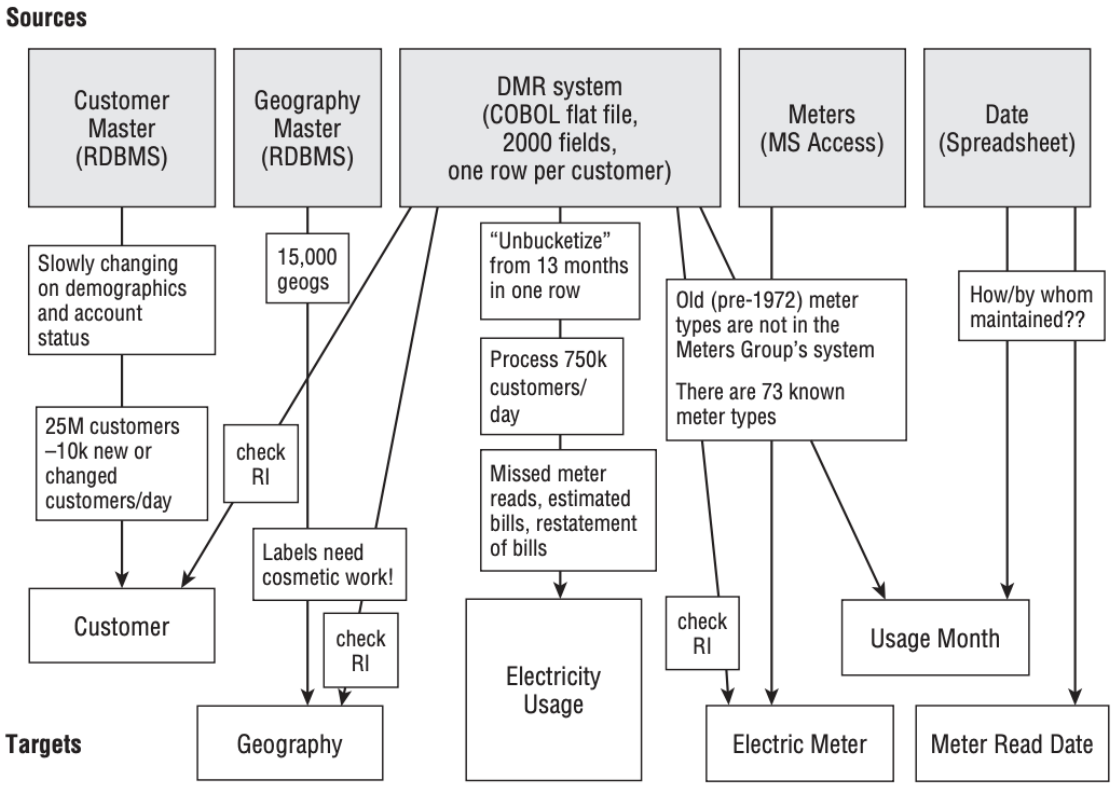
systémy mají své důvody a tím jsou v prvé řadě rozdílné požadavky jak transakčního systému, tak analytického systému, které by nebylo možné naplnit. To vychází z různého uspořádání dat, kde není úplně jednoduché provádět stejné dotazy nad ER diagramem jak v hvězdicovém schématu. Navíc OLTP systém je zaměřen na rychlou odezvu a atomicitu operaci, kde naopak OLAP systém je zaměřen zpracování velkého množství dat s vyšší odezvou.

3.6.4 ETL

ETL (Extract Transform Load) proces je velkou skrytou součástí vývoje data-mining nebo Business Intelligence aplikací. Skládá se ze tří kroků, jejichž cílem je získání dat z různých zdrojů, jejich úprava do požadovaného formátu a nahrání do cílového úložiště.

3.6.4.1 Extrakce

Extrakce dat je prvním krokem v tomto procesu. Systém provádějící extrakci musí být připraven na vytáhnutí dat v různých formátech z různých zdrojů dat. V tom nejlepším případě jsou data dostupná rovnou v jednom systému v nějakém dobře známém a jednoduše zpracovatelném formátu. Častější bývá situace, kdy data je potřeba vytáhnout z více zdrojů v různých formátech. Je třeba si ujasnit odkud se která data budou brát jako např. na obrázku č. 4.



Obrázek 10 - Příklad plánu zdrojů dat
Zdroj: Kimball (8)

Při přesunu se dají použít dvě hlavní metody, jako soubor nebo proud (angl. stream). Zatímco při použití proudu je možné všechny tři kroky provést zároveň v jednom procesu, tak při použití souboru se jedná o čtyři diskrétní kroky (extrakce do souboru, přesun souboru, transformace dat a načtení transformovaných dat do úložiště). Ačkoliv se použití proudu může zdát jako výhodnější řešení, tak extrahování do souboru má své výhody. Je jednodušší celý proces provést znovu (za předpokladu, že původní datový soubor zůstal nezměněn) bez jakéhokoliv dopadu na zdrojový systém. Soubor lze zašifrovat vlastním algoritmem a provést kompresi před transferem dat po síti. Lze i jednoduše ověřit integritu dat pomocí hashovací funkce.

Pokud se data neextrahují jednorázově, je potřeba zařídit inkrementální extrakci dat. To znamená, že se již neextrahují data, která jsou již zpracovaná a uložená v datovém skladu. Tím by mohlo docházet k duplicitám a zbytečnému vytížení výpočetní a úložné kapacity. (8)

3.6.4.2 Transformace

Poté co jsou data vyextrahována do dočasného úložiště (angl. staging area), existuje mnoho potencionálních transformací jako je například čištění dat. To spočívá odstraňování chyb, které mohou být různého druhu. Nejčastější jsou chybějící hodnoty, neplatné hodnoty nebo rozdílný formát dat. V tomto kroku je třeba ošetřit veškeré chyby, které by později mohli mít vliv na kvalitu.

Při chybějících hodnotách (NULL) ve faktových tabulkách je v pořádku, jelikož agregátní funkce (suma, počet, minimum, maximum, průměr a další) jsou schopny tyto hodnoty vynechat. Problém nastává při chybějících hodnotách u cizích klíčů faktových tabulek. Tuto situaci si lze představit na objednávce u které neznáme zákazníka. Obecně by se mělo vyhnout NULL hodnotám, protože ty nemají jasně určený význam a může nastat více situací kdy může NULL hodnota vzniknout a poté již není možné mezi případy rozlišit a dohledat jejich význam. Při chybějící hodnotě lze je nahradit například řetězcem „--missing--“ a při chybějícím cizím klíči jej lze nahradit například řetězcem „--ref-integity-error--“ tak, aby bylo možné zpětně dohledat objednávky bez zákazníka.

Důležitý je i převod datových typů, tak aby se s daty dalo dále pracovat. Například časové údaje je třeba převést na nativní datový typ zvoleného nástroje, tak aby mu rozuměl a byl schopen s nimi pracovat. Situace je stejné pro čísla a další speciální datové typy. Pokud se tento krok provede správně tak se práce dále při zpracování dat jednodušší.

Dimenze dat, jak již bylo zmíněno můžou pocházet z více zdrojových dat. Při kombinování více zdrojů může narážet na problémy, že primární klíče mohou být rozdílné, ovšem se stejným významem. Řetězec „IBM“ značí stejnou společnost jako „I.B.M“. S těmito nepřesnostmi je třeba se vypořádat.

3.6.4.3 Nahrání dat

Poté co jsou data transformována do konečné podoby, tak je třeba je nahrát ve výsledné formě do úložiště (datového skladu), ze kterého budou dále analyzována a prezentována. Jedná se o přesun dat z dočasného úložiště, kam byla data ukládány během transformace do trvalého úložiště. (8)

3.6.5 ELT vs. ETL

ELT (Extract Load Transform) proces se oproti ETL liší zásadně v pořadí operací, kdy transformace probíhá až po uložení dat. To znamená, že ukládána jsou všechna data ve formátu tak jak byla extrahována v původním formátu. Tento přístup přináší vyšší flexibilitu, ale vyšší nároky na výpočetní kapacitu. Úložiště takovýchto nestrukturovaných dat se nazývá datové jezero (ang. Data Lake). Jelikož jsou ukládána všechna data, tak nároky na úložiště jsou řádově vyšší. To je vykoupeno možností provádět transformace libovolně za běhu ze surových dat. To se hodí v případě, kdy transformace byla provedena nesprávně a je třeba celý proces zopakovat. V ELT je tato chyba snadněji opravitelná, jelikož veškeré transformace se dějí za běhu, tak chyba nemá žádné větší důsledky.

3.7 Praktická část

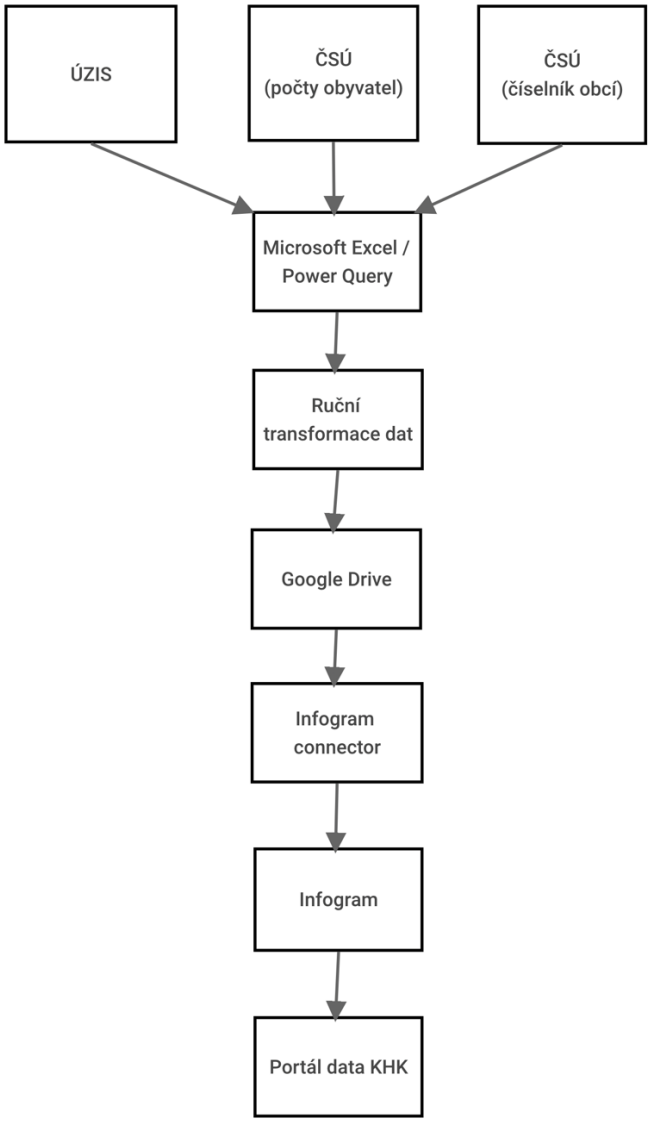
3.7.1 Portál Data KHK

Současný portál DataKHK (https://www.datakhk.cz/) je dílem datového týmu Královéhradeckého Kraje. Portál se dělí do aplikací, dashboardů a infografik. Aplikace zobrazují data promítnutá do mapových podkladů. Dashboardy zobrazují přehlednou statistiku jedné oblasti v několika grafech, tak aby čtenář jednoduše nabyl znalosti dat. Infografiky jsou výstupem analýz zpracovaných pro Královéhradecký kraj a jsou v nich popsány výsledky analýzy.

Samotný portál se popisuje následovně: „Všechna volně přístupná data Královéhradeckého kraje na jednom místě. Tak by se dalo nazvat motto tohoto portálu. Jde o centralizované informační místo pro veřejnost. Shromažďují se zde data z řady zdrojů. Jedná se zejména o data kraje, statistická data a otevřená data. Následně se zpracovávají a transformují do uživatelsky přívětivých formátů veřejnosti např. ve formě jednoduchých infografik, vyhodnocených statistik vzájemně provázaných dat, ročních reportů a trendů či datového katalogu ve formátu open dat. Jsou podkladem pro mapové výstupy, pro tiskové zprávy, sociální sítě, reportáže a další. Díky webové formě je datový portál k dispozici všem 24 hodin denně, 7 dní v týdnu, bez nutnosti zdlouhavého vyhledávání statistik pro veřejnost.“ (12).

Jak již bylo zmíněno v úvodu, praktická část se zabývá možnostmi automatizace zpracování dat pro užití v datovém portále. Před tím, než je možné proces automatizovat, je třeba se podívat na stávající procesy.

3.7.2 Aktuální infrastruktura portálu Data KHK

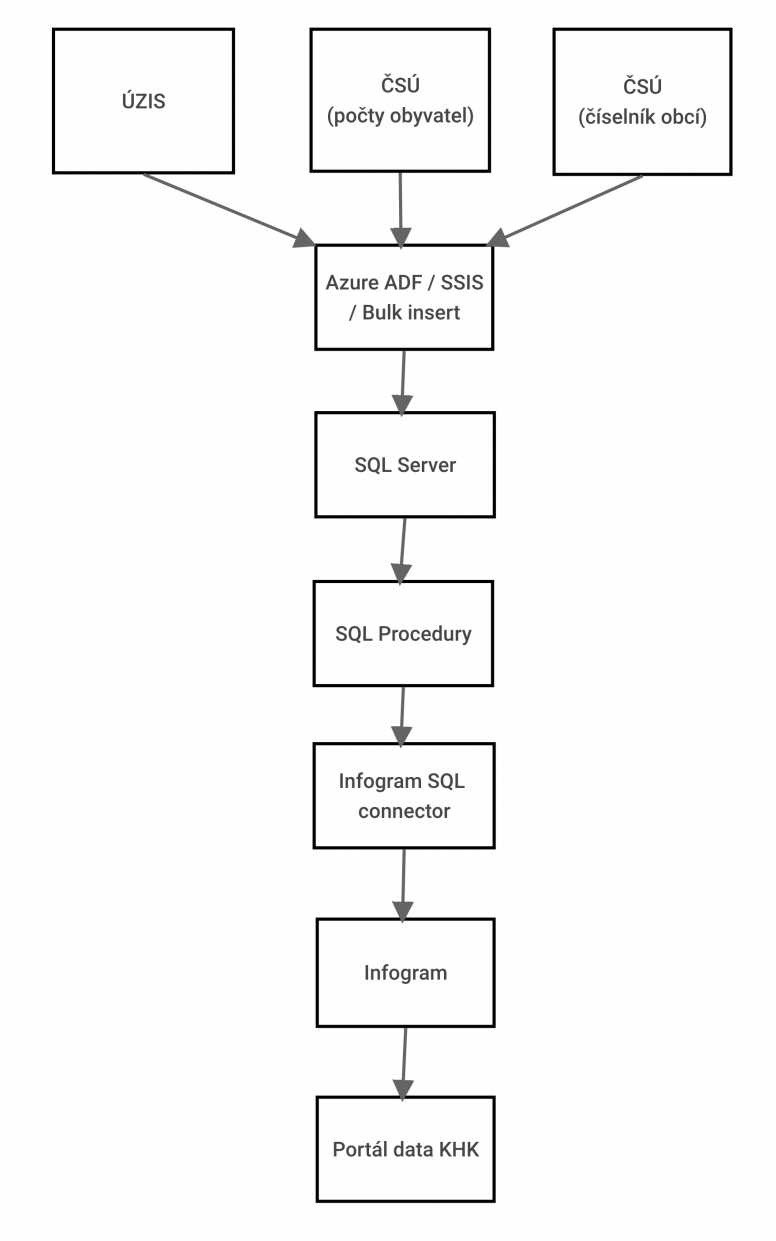


Obrázek 11 - Datová infrastruktura Data KHK
Zdroj: vlastní tvorba

Na obrázku č. 11 lze vidět aktuální proces zpracování dat. Prvním krokem je lokalizace zdrojových dat například v Národním katalogu otevřených dat nebo na jednotlivých ministerstvech a úřadech. Dále je potřeba tyto data stáhnout a ručně upravit tak, aby obsahovala všechny informace potřebné pro vizualizace. Jakmile jsou data v požadovaném formátu, tak se nahrají do cloudového úložiště Google Disk. Platforma Infogram již obsahuje konektor, který je s Google Diskem kompatibilní a z dat je možné tvořit vizualizace.

Například pro přehled počtu osob s uděleným pobytovým oprávněním v souvislosti s válkou na Ukrajině je tento proces opakovat na denní bázi. To znamená celý proces vyobrazený na obrázku č. 11 opakovat denně.

3.7.3 Návrh automatizované struktury



Obrázek 12 - Návrh automatizovaného procesu
Zdroj: Vlastní tvorba

Návrh automatizovaného spočívá v nasazení ELT procesu do prostředí Data KHK. Tento proces začíná podobně jako u stávajícího procesu u lokalizace potřebných dat na jednotlivých ministerstvech a úřadech. Tato data jsou poté načtena do vstupních tabulek databáze pomocí některé z metod uvedených v dalších kapitolách. Následně jsou data rozdělena do jednotlivých faktových a dimenzionálních tabulek. Pro jednotlivé potřeby vizualizace jsou na výstupu vytvořeny tabulky s agregovanými informacemi pro potřeby konkrétní vizualizace.

3.7.4 Výběr technologií

Výběr technologií byl dán především možnostmi napojení na platformu Infogram. Ta podporuje import z úložišť Google Disk, Dropbox nebo import JSON feedu z URL adresy.

Tohoto napojení pomoci JSON feedu by se dalo využít například napsáním aplikace, které bude potřebná data stahovat, ukládat a transformovat a dále předávat Infogramu pomocí REST API. Ovšem napsání takové aplikace by vyžadovalo obrovské úsilí, komplikovanou infrastrukturu a tým složený z vývojářů, analytiků a databázových specialistů.

Infogram poskytuje i napojení přímo na datové úložiště (MySQL, PostgreSQL, Amazon Redshift, Oracle nebo Microsoft SQL Server) pomocí jednoduchého SQL dotazu. Z vyjmenovaných úložišť byl vybrán Microsoft SQL Server, neboť tým Data KHK má možnost provozovat a udržovat vlastní instanci SQL Serveru a tým ovládající jazyk SQL.

3.7.5 Automatizovaný procesu na konkrétním případu

Pro účely demonstrace proveditelnosti automatizace byla vybrána data a vizualizace související s počtem provedených očkování. Tato data pocházejí od Ústavu zdravotnických informací a statistiky ČR. Byly definovány následující vizualizace:

1. Celkový počet aplikovaných dávek vakcíny po okresech Královéhradeckého kraje
2. Celkový počet aplikovaných dávek vakcíny po okresech Královéhradeckého kraje a po pořadí dávky
3. Přehled celkového počtu aplikovaných dávek po okresech Královéhradeckého kraje a věkové skupině

Pro následující přehledy je třeba sehnat data o počtu vykázaných očkováních, okresech v Královéhradeckém kraji a počtu obyvatel v jednotlivých okresech Královéhradeckého kraje. Tato data lze získat například z Portálu otevřených dat ČR (https://data.gov.cz). Pro potřeby vizualizací byly použity následující datasety:

1. Data o vykázaných očkováních - COVID-19: Základní přehled vykázaných očkování (13)
2. Číselník obcí s rozšířenou působností – Číselník ORP (14)
3. Počty obyvatel jednotlivých ORP – Získáno z dat týmu Data KHK
4. Vztahy okresů k ORP – Získáno z dat týmu Data KHK

3.7.6 Zdroje dat a analýza

První dataset o vykázaných očkováních je extrahován z portálu Onemocnění aktuálně patřící pod Ministerstvo Zdravotnictví České republiky. Pro potřeby vizualice se používá dataset „COVID-19: Základní přehled vykázaných očkování“. Ten má následující strukturu:

Název sloupce	**Datový typ**	**Popis**
id	string	
kraj_nazev	string	Název kraje, ve kterém se nachází očkovací místo.
kraj_nuts_kod	string	Identifikátor kraje podle klasifikace NUTS 3, ve kterém se nachází očkovací místo.
orp_bydliste	string	Bydliště (na úrovni názvu obce s rozšířenou působností) očkované osoby.
orp_bydliste_kod	string	Bydliště (na úrovni kódu obce s rozšířenou působností) podle číselníku ÚZIS ČR (https://pzu-api.uzis.cz/api/orp) očkované osoby.
vakcina	string	Název očkovací látky.
vakcina_kod	string	Kód očkovací látky podle klasifikace ÚZIS ČR.

poradi_davky	integer	Pořadí dávky (první, druhá, posilující) danou očkovací látkou.
vekova_skupina	string	Rozdělení očkovaných osob podle věku do skupin: 0-17, 18-24, ..., 75-79, 80+, nezařazeno - pokud nelze určit věkovou skupinu.
pohlavi	string	V případě neuvedení pohlaví u vykázaného očkování se jedná o cizince, u kterého není možné prostřednictvím centrálních registrů pohlaví identifikovat.
pocet_davek	integer	Počet vykázaných dávek očkování.

Obrázek 13 - Struktura přehledu očkování
Zdroj: https://onemocneni-aktualne.mzcr.cz/api/v2/covid-19/ockovani-zakladni-prehled.csv-metadata.json

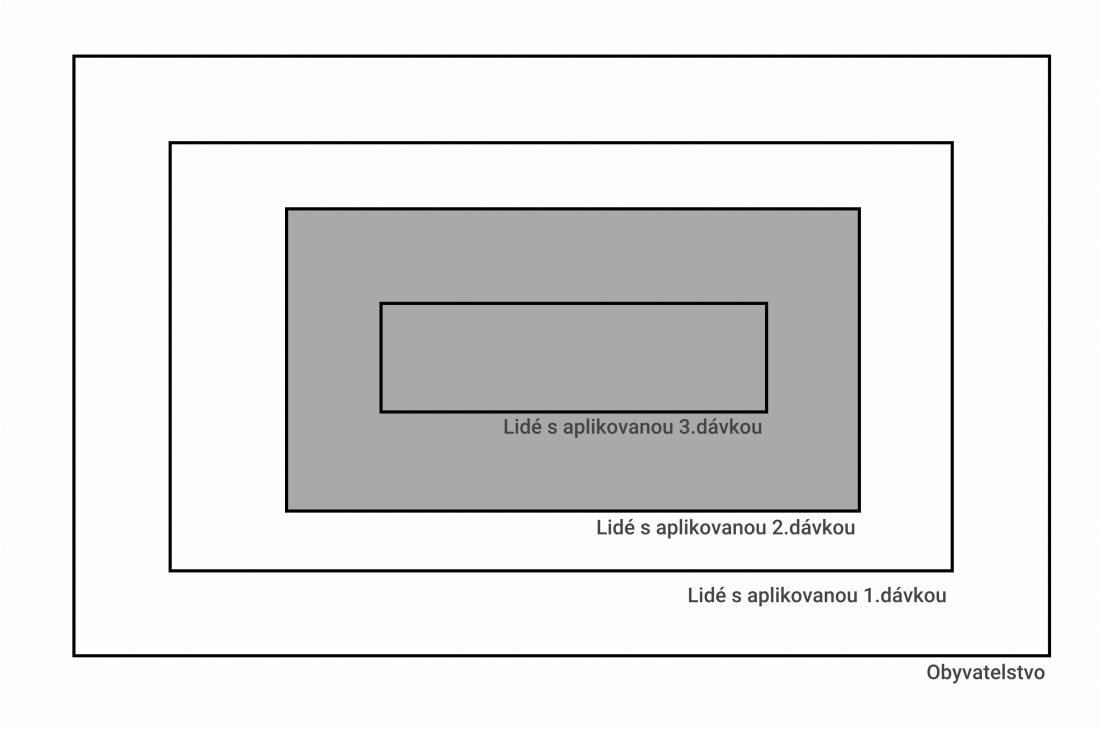
Ze struktury lze vyčíst, že obsahuje téměř vše potřebné. Sloupce „kraj_nazev“ a „kraj_nuts_kod“ neudávají kraj trvalého bydliště očkované osoby, ale kraj, ve kterém bylo očkování provedeno a v datech lze tudíž najít například, záznam o čtyřech provedených očkování v Hlavním městě Praha ve skupině mužů ve věku 25-29 let s trvalým bydlištěm v obci s rozšířenou působností Slavkov u Brna. Jelikož analýza má za cíl získat přehled o očkování obyvatelů Královéhradeckého kraje, tak tyto dva sloupce nejsou vůbec potřeba.

Sloupce „orp_bydliste“ a „orp_bydliste_kod“ údávají název a kód z číselníku CISORP (Číselník správních obvodů obcí s rozšířenou působností).

Dataset neobsahuje okres očkovaného, ovšem to je možné zjistit právě díky kódu obce s rozšířenou působností.

Sloupce „vakcina“ a „vakcina_kod“ úvádí název očkovací látky a její kód. Pro potřeby stanovených cílů nejsou tyto sloupce potřeba, ovšem pro potřeby datového skladu je výhodné je zpracovávat pro budoucí rozšíření.

Sloupec „poradi_davky“ udává číslo aplikované dávky u skupiny. U tohoto údaje je třeba se zastavit a důkladně pochopit jeho význam.



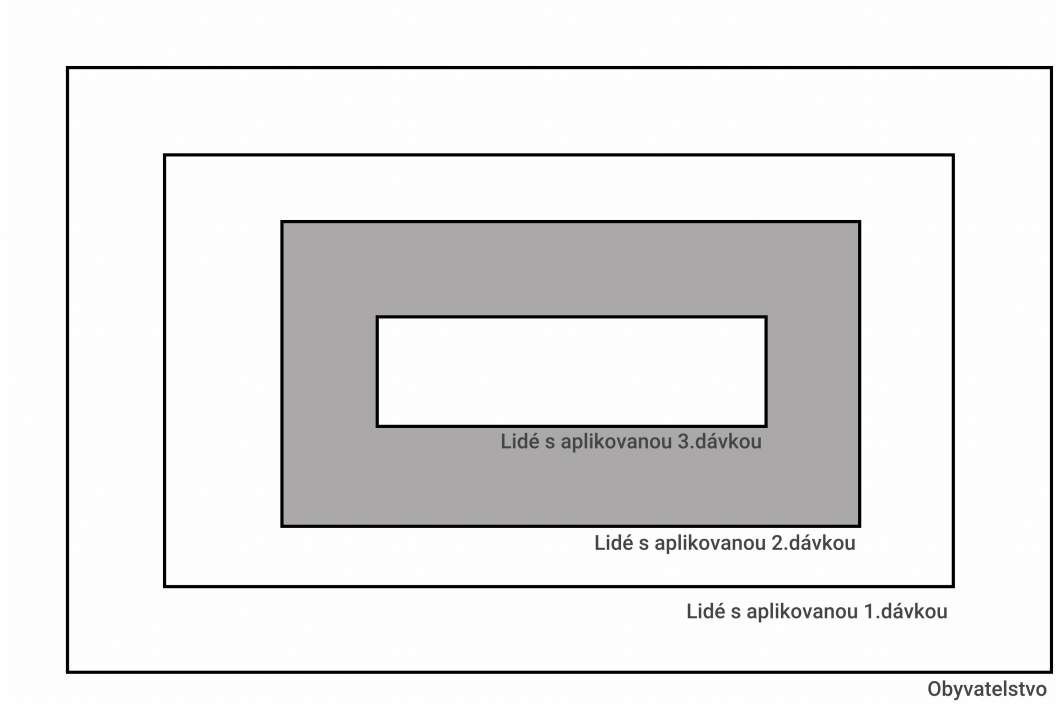
Obrázek 14 - Vysvětlující diagram sloupce "poradi_davky"
Zdroj: Vlastní tvorba

Například pokud je u skupiny uvedeno v tomto sloupci číslo 2, tak to znamená, že lidé v této skupině mají aplikovanou alespoň druhou dávku (ve skutečnosti tedy mají aplikovanou alespoň druhou, třetí nebo čtvrtou dávku). To znamená, že dataset neobsahuje počty lidí, ovšem počty aplikovaných dávek. Pokud tato čísla sečteme, tak získáme celkový počet aplikovaných dávek, nikoliv

počet obyvatelů s daným očkováním. Toto číslo se dá charakterizovat následujícím vzorcem:

$$D_n = \sum_{i=n}^{m} D_i$$

V tomto vzorci D značí pořadí dávky, n značí pořadí dávky z datasetu a m je nejvyšší možná dávka.



Obrázek 15 – Diagram reprezentující obyvatelstvo s právě dvěmi dávkami
Zdroj: Vlastní tvorba

Pokud bychom chtěli dostat počet lidí s právě druhou dávkou jako na obrázku č. 15, tak je třeba od aktuální skupiny odečíst počet lidí s aplikovanou o jednu více davkou.

$$d_n = D_n - D_{n+1}$$

Sloupec věková skupina udává věk skupiny obyvatelstva po pětiletých intervalech s vyjímkou skupiny 0 až 17 let (ta byla zavedena nedávno a vznikla sloučením skupin 0 až 11 let, 12 až 15 let a 16 až 17 let) a skupiny 80 let a více (80+).

Aby bylo možné počítat procentní proočkovanost v okresech a krajích, tak je třeba znát celkový počet obyvatel. Tyto údaje nejsou v datasetu obsaženy, nicméně je možné díky kódu obce s rozšířenou působností nebo kódem okresu data propojit s jiným zdrojem. Proto byla použita data s následující strukturou:

Název sloupce	Datový typ	Popis
okres_nazev	string	Název okresu
orp_nazev	string	Název obce s rozšířenou působností
orp_kod	string	Kód obce s rozšířenou působností z číselníku CISORP
okres_kod	string	Kód okresu z číselníku „OKRES_LAU“
vekova_skupina	string	Věková skupina
pocet_obyvatel	integer	Počet obyvatel v obci s rozšířenou působností a věkové skupině

Tabulka 6 - Struktura datasetu počtu obyvatel
Zdroj: Datový tým KHK

3.7.7 Implementace automatizovaného procesu

Jak již bylo uvedeno v předchozích kapitolách, jako primární úložiště dat byl zvolen SQL Server. Pro potřeby čištění dat, spojování datasetů a normalizace do hvězdicového schématu byly zvolena čtyřvrstvá architektura kde v první vrstvě nazvané „00“ se vytvoří tabulky a jsou naplněny surovými daty. V další vrstve „10“ se data spojí a vytvoří se potřebné kombinace dat. Ve vrstvě „20“ se data rozčlení do hvězdicového schématu a v poslední vrstvě „30“ se sestavují pohledy na data, tak aby reprezentovali potřebné vizualizace.

3.7.8 Vrstva „00“

```
DROP TABLE IF EXISTS [00_prehled];

CREATE TABLE [00_prehled] (
    id nvarchar(300),
    kraj_nazev nvarchar(300),
    kraj_nuts_kod nvarchar(300),
    orp_bydliste nvarchar(300),
    orp_bydliste_kod nvarchar(300),
    vakcina nvarchar(300),
    vakcina_kod nvarchar(300),
    poradi_davky nvarchar(300),
    vekova_skupina nvarchar(300),
    pohlavi nvarchar(300),
    pocet_davek nvarchar(300),
);

BULK INSERT
    [00_prehled]
FROM 'C:/ockovani-zakladni-prehled.csv'
WITH (
    FORMAT = 'CSV',
    FIELDTERMINATOR = ',',
    ROWTERMINATOR = '\n',
    FIRSTROW = 2
)
```

Načtení pomocí příkazu BULK INSERT se nepodařilo zprovoznit, proto se nyní používá služba Azure Data Factory, která umí pravidelně (každých 24 hodin) stáhnout CSV soubor z HTTP URL a data vložit do cílové tabulky „00_prehled“. Pokud by při nasazení nebyla použita služba Azure, tak je zde i možnost data vkládat pomocí následujícího skriptu napsaném v jazyce PHP:

```php
<?php

$serverName = '';
$connectionInfo = ['database' => '', 'username' => '', 'password'
=> ''];

$url = 'https://onemocneni-aktualne.mzcr.cz/api/v2/covid-
19/ockovani-zakladni-prehled.csv';

$session = curl_init($url);
$save = './prehled.csv';

$csvFilePath = fopen($save, 'wb');
curl_setopt($session, CURLOPT_FILE, $csvFilePath);
curl_setopt($session, CURLOPT_HEADER, 0);
curl_exec($session);
curl_close($session);
fclose($csvFilePath);

try {
    $conn = new PDO(

"sqlsrv:Server=$serverName;Database={$connectionInfo['database']}
",
        $connectionInfo['username'],
        $connectionInfo['password']
    );
    $conn->setAttribute(PDO::ATTR_ERRMODE,
PDO::ERRMODE_EXCEPTION);
    if (($handle = fopen($csvFilePath, "r")) !== FALSE) {
        $stmt = $conn->query('DELETE FROM [00_prehled]');
        while (($data = fgets($handle)) !== FALSE) {
            $rowData = str_getcsv($data);

            $sql = "INSERT INTO [00_prehled] (
                id, kraj_nazev, kraj_nuts_kod,
                orp_bydliste, orp_bydliste_kod,
                vakcina, vakcina_kod, poradi_davky,
                vekova_skupina, pohlavi, pocet_davek
            ) VALUES (?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?)";
            $stmt = $conn->prepare($sql);
            foreach (range(1, 11) as $index) {
                $stmt->bindParam($index, $rowData[$index]);
            }
            $stmt->execute();
        }
        fclose($handle);
    }
    $conn = null;
} catch (PDOException $e) {
    echo 'Error: ' . $e->getMessage();
}
```

Pro dataset o počtech obyvatel probíhá podobný proces, ovšem s tím rozdílem, že tato data se nemění tak často, proto postačuje je vložit jednou bez pravidelné aktualizace.

```
DROP TABLE IF EXISTS [dbo].[00_pocet_obyvatel];

CREATE TABLE [dbo].[00_pocet_obyvatel]
(
    okres_nazev    nvarchar(100),
    orp_nazev      nvarchar(100),
    orp_kod        nvarchar(100),
    okres_kod      nvarchar(100),
    vekova_skupina nvarchar(100),
    pocet_obyvatel nvarchar(100)
);

INSERT INTO [00_pocet_obyvatel] (okres_nazev, orp_nazev, orp_kod,
okres_kod, vekova_skupina, pocet_obyvatel)
VALUES (N'Náchod', N'Broumov', N'5201', N'CZ0523', N'0-17', N'2
827');
...
INSERT INTO [00_pocet_obyvatel] (okres_nazev, orp_nazev, orp_kod,
okres_kod, vekova_skupina, pocet_obyvatel)
VALUES (N'Trutnov', N'Vrchlabí', N'5215', N'CZ0525', N'80+', N'1
124');
```

Tím jsou data načtena do databáze v surové formě a lze pokračovat s jejich dalším zpracováním.

3.7.9 Vrstva „10“

V první řadě je třeba převést sloupce na správné datové typy (například ve vrstvě 00 je počet obyvatel typem nvarchar(100)) a je třeba jej převést na číslo (integer)

```
INSERT INTO [10_pocet_obyvatel]
SELECT
    okres_nazev,
    orp_nazev,
    orp_kod,
    okres_kod,
    vekova_skupina,
    CAST(REPLACE(pocet_obyvatel, ' ', '') AS int)
FROM
    [00_pocet_obyvatel]
```

Stejný proces je proveden u datasetu s počty provedeného očkování, ovšem zde je potřeba zařídit více věcí:

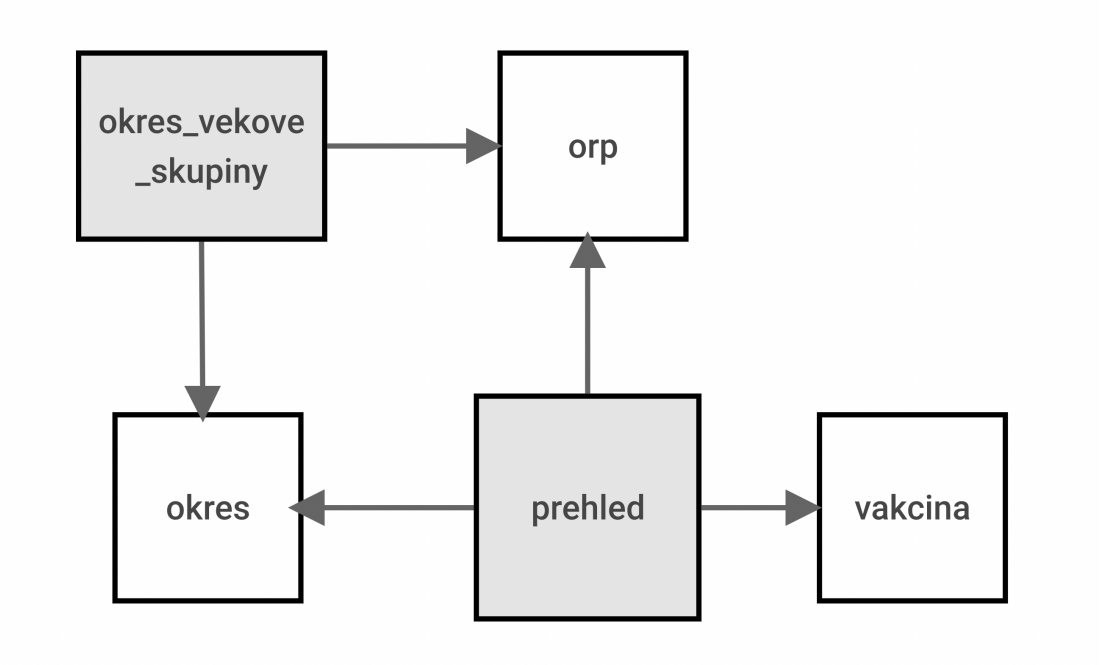
```
INSERT INTO [10_prehled]
SELECT
    p.orp_bydliste_kod AS orp_kod,
    p.vakcina,
    p.vakcina_kod,
    CAST(p.poradi_davky AS int) AS poradi_davky,
    p.vekova_skupina,
    p.pohlavi,
    p.pocet_davek AS pocet_davek,
    (SELECT TOP 1 orp.okres_kod FROM [00_pocet_obyvatel] orp
WHERE orp.orp_kod = p.orp_bydliste_kod) as okres_kod
FROM
    (
        SELECT
            i.orp_bydliste_kod, i.vakcina_kod, i.vakcina,
i.poradi_davky, i.vekova_skupina,
            i.pohlavi, SUM(CAST(i.pocet_davek AS int)) AS
pocet_davek
        FROM [00_prehled] i
        GROUP BY orp_bydliste_kod, vakcina_kod, vakcina,
poradi_davky, vekova_skupina, pohlavi
    ) p
WHERE
    p.orp_bydliste_kod IN (
        '5201', '5202', '5203', '5204', '5205',
        '5206', '5207', '5208', '5209', '5210',
        '5211', '5212', '5213', '5214', '5215'
    )
```

Převádí se zde datové typy, odfiltrují se zde obce s rozšířenou působností mimo Královéhradecký kraj a ještě k tomu se odstraní sloupce „kraj_nazev“ a „kraj“. To nelze provést jednoduchým vynecháním sloupců z výrazu SELECT, neboť by došlo k duplikaci řádků. Skupina by se zde mohla vyskytovat vícekrát, protože část skupiny byla naočkována např. v Jihomoravském kraji a druhá

v Olomouckém kraji. Tento údaj již není relevantní a proto je potřeba oba záznamy sečíst.

3.7.10 Vrstva „30“

V další vrstvě „30“ se data extrahují do faktu a dimenzí dle následujícího schématu:



Tabulka 7 - Diagram schématu faktů a dimenzí
Zdroj: vlastní tvorba

Šedě podbarvené obdélníky reprezentují dimenzionální tabulky. V tomto případě je zapotřebí dvou tabulek, neboť základní přehled očkování reprezentuje skupiny členěné dle věku, trvalého bydliště, pohlaví a typu použité vakcíny, zatímco demografické skupiny jsou členěny pouze podle trvalého bydliště a věku.

Zároveň se zde probíhá přepočet z celkového počtu vyočkovaných dávek na počet obyvatel ve skupině s právě daným číslem dávky. Toho je docíleno pomocí funkce LAG s klauzulí OVER. Tato funkce pracuje tak, že vrací hodnotu z předchozí skupiny, která je definovaná ve funcki OVER.

```
INSERT INTO [dbo].[20_prehled]
    SELECT
        orp_kod,
        vakcina_kod,
        poradi_davky,
        vekova_skupina,
        pohlavi,
        pocet_davek,
        ABS(pocet_davek - LAG(pocet_davek, 1, 0) OVER (
            PARTITION BY
                orp_kod,
                vakcina_kod,
                vekova_skupina,
                pohlavi
            ORDER BY poradi_davky DESC)) AS pouze_davka,
        okres_kod,
        orp_vek_pocet_obyvatel
    FROM
        [10_prehled] p
```

Stejný proces se děje pro druhou dimenzionální tabulky s počty obyvatel v okresech a obcích s rozšířenou působností. Pro samotné faktové tabulky stačí vybrat unikátní hodnoty pomocí výrazu statement následovně:

```
INSERT INTO [dbo].[20_vakcina]
SELECT DISTINCT vakcina_kod, vakcina FROM [00_prehled]

INSERT INTO [dbo].[20_orp]
SELECT DISTINCT
   orp_kod, orp_nazev
FROM [00_pocet_obyvatel]

INSERT INTO [dbo].[20_okres]
SELECT DISTINCT
   okres_kod, okres_nazev
FROM [00_pocet_obyvatel]
```

Tímto je vrstva 20 kompletní a je transformována do hvězdicového schématu s vyčištěnými daty, proto je vytvářet samotné pohledy na data ve vrstvě „30“.

Pro pohled proočkovanosti dle okresů a dávky je nutné vypočítat i počet obyvatel jednotlivých okresů bez jakékoliv očkování následovně:

$$D_0 = P_{Okres} - \sum_{i=1}^{m} D_i$$

Kde D_0 je počet obyvatel bez očkování, P_{okres} je počet obyvatel v okrese a D_i značí počet obyvatel v okrese práve s i dávkami. Takto lze vypočíst velikost skupiny obyvatel bez očkování. V přehledu je značena jako skupina s nultým pořadím dávky.

```sql
CREATE OR ALTER VIEW [dbo].[30_prehled_proockovanost] AS
(
    SELECT
        SUM(pocet_davek) / CAST(
            (
                SELECT
                    SUM(pocet_obyvatel) FROM [20_okres] [i]
                WHERE [i].okres_kod = [p].okres_kod
            ) AS float
        ) AS pocet_davek_procent,
        SUM(pocet_davek) AS pocet_davek,
        [p].poradi_davky,
        [o].okres_nazev
    FROM [dbo].[20_prehled] [p]
    JOIN [dbo].[20_okres] [o] ON [p].okres_kod = [o].okres_kod
    GROUP BY [p].poradi_davky, [p].okres_kod, [o].okres_nazev
) UNION (
    SELECT
        1 - (SUM([p].pocet_davek) / CAST((
            SELECT
               SUM([i].pocet_obyvatel)
            FROM [20_okres] [i]
            WHERE [i].okres_kod = [p].okres_kod
        ) AS float)) AS pocet_davek_procent,
        (CAST((
            SELECT
               SUM([i].pocet_obyvatel)
            FROM [20_okres] [i]
            WHERE [i].okres_kod = [p].okres_kod
        ) AS float)) - SUM([p].pocet_davek) AS pocet_davek,
        0 AS poradi_davky,
        [o].okres_nazev
    FROM [20_prehled] [p]
    JOIN [dbo].[20_okres] [o] ON [p].okres_kod = [o].okres_kod
    WHERE poradi_davky = 1
    GROUP BY [p].poradi_davky, [p].okres_kod, [o].okres_nazev
)
```

Přehled proočkovanosti po okrese je o poznání jednodušší, neboť není potřeba dopočítávat obyvatele bez očkování. Je dostačující vydělit počet obyvatelů okresu počtem obyvatel s alespoň jednou dávkou.

```
CREATE OR ALTER VIEW [dbo].[30_prehled_proockovanost_okresy] AS
(
    SELECT
        SUM(pocet_davek) / CAST(
            (
                SELECT
                    SUM(pocet_obyvatel) FROM [20_okres] [i]
                WHERE [i].okres_kod = [p].okres_kod
            ) AS float
        ) AS pocet_davek_procent,
        SUM(pocet_davek) AS pocet_davek,
        [o].okres_nazev
    FROM [dbo].[20_prehled] [p]
    JOIN [dbo].[20_okres] [o] ON [p].okres_kod = [o].okres_kod
    WHERE poradi_davky = 1
    GROUP BY [p].poradi_davky, [p].okres_kod, [o].okres_nazev
)
```

Přehled proočkovanosti po okresech a věkových skupinách je podobný, ovšem s rozdílem, že data jsou zobrazena ještě dle věkové skupiny:

```
CREATE OR ALTER VIEW [dbo].[30_prehled_proockovanost_vek_skupiny]
AS
(
    SELECT
            SUM(pocet_davek) / CAST(
                (
                    SELECT
                        SUM(pocet_obyvatel) FROM
[20_okres_vekove_skupiny] [i]
                    WHERE
                            [i].okres_kod = [p].okres_kod
                      AND [i].vekova_skupina = [p].vekova_skupina
                ) AS float
            ) AS proockovanost,
            SUM(pocet_davek) AS pocet_davek,
            [p].vekova_skupina,
            [o].okres_nazev
    FROM [dbo].[20_prehled] [p]
             JOIN [dbo].[20_okres] [o] ON [p].okres_kod =
[o].okres_kod
    WHERE poradi_davky = 1
    GROUP BY [p].vekova_skupina, [p].okres_kod, [o].okres_nazev
)
```

3.7.11 Napojení datového skladu na Infogram

Napojení na vizualizační platformu Infogram je otázkou správného nastavení adresy a přihlašovacích údajů k SQL Serveru. Poté je třeba již vložit správný dotaz na pohled z vrstvy 30 např. následovně:

```
SELECT * FROM [dbo].[30_prehled_proockovanost_vek_skupiny]
```

3.7.12 Struktura projektu

Projekt je členěn do samostatných SQL příkazů do složek podle vrstev, tak aby bylo možno jednotlivé vrstvy přehledně upravovat. Zároveň jsou v kořenové složce dva skripty. První pro načtení dat do tabulky „00_prehled“ a druhý pro spojení všech SQL příkazů do jedné procedury, která se může automatizovaně spouštět jako procedura.

```php
<?php

$scripts = concatFolderContents('./*');

$procedure = <<<SQL
IF EXISTS (
    SELECT * FROM sys.objects
   WHERE type = 'P' AND OBJECT_ID =
OBJECT_ID('UpdateDataProcedure'))
BEGIN
    DROP PROCEDURE UpdateDataProcedure
END

CREATE PROCEDURE UpdateDataProcedure
AS
BEGIN
     $scripts
END
SQL;

file_put_contents('./sp.sql', $procedure);


function concatFolderContents(string $folderPath) {
    $files = glob(rtrim($folderPath, '/') . '/*.sql');

    return implode(
        PHP_EOL . PHP_EOL,
        array_map(function (string $file): string {
            return sprintf(
                '-- %s %s %s',
                $file,
                PHP_EOL,
                file_get_contents($file),
            );
        }, $files)
    );
}
```

4 Shrnutí výsledků

Po konzultacích s datovým týmem Královéhradeckého kraje bylo zanalyzováno stávající řešení a proces jakým jsou data zpracovávána. Na základě této analýzy byly identifikovány kroky v procesu zpracování dat, které je možné zautomatizovat. Pro počáteční fázi implementace byly zvoleny následující tři již existující vizualizace, které již existují a zdrojová data berou z Google Disku kam se vkládají ručně:

1. Celkový počet aplikovaných dávek vakcíny po okresech Královéhradeckého kraje
2. Celkový počet aplikovaných dávek vakcíny po okresech Královéhradeckého kraje a po pořadí dávky
3. Přehled celkového počtu aplikovaných dávek po okresech Královéhradeckého kraje a věkové skupině

Cílem práce bylo navrhnout datový sklad tak aby, bylo možné automatizovaně zpracovávat data pro vizualizace.

Datový sklad je plně schopen uchovávat a automatizovaně zpracovávat data o vyočkovaných dávkách vakcíny proti nemoci COVID-19. Data pro všechny tři vizualizace jsou dostupná jako pohled ve vrstvě 30 (anglicky view) pro napojení na vizualizační platformu Infogram, která je dále využita na datovém portále KHK.

5 Závěry a doporučení

Implementací datového skladu není plně slněn cíl automatizace, neboť je třeba datový sklad nasadit do reálného provozu na vlastní hardware Královéhradeckého kraje, což je mimo rozsah této práce. To obnáší instalaci SQL Serveru a automatického spouštění skriptu (ať již navrhovaným PHP skriptem load.php či jiným řešením) pro plnění dat do tabulky 00_prehled a následným spouštěním procedury UpdateDataProcedure pro spouštění přepočtu dat.

Dále je nutné po určitou dobu sledovat, zda se výstupní data shodují s daty, která jsou výsledkem stávajícího procesu a případné chyby identifikovat. Ty mohly nastat jak při analýze, tak i při implementaci.

Pokud se navrhované řešení projeví jako spolehlivé, tak je možné zapojit další data a vizualizace. S rostoucím počtem zpracovávaných dat bude klesat i náročnost na zapracování nových dat, jelikož již nebude nutné zpracovávat např. počty obyvatel jednotlivých okresu.

Mezi největší nedostatky řešení patří nutnost ručně aktualizovat data o počtu obyvatel okresů, čehož je možné docílit nalezením vhodného datasetu, který udává počet obyvatel jednotlivých okresů po věkových skupinách a jeho automatickou aktualizací do tabulky 00_pocet_obyvatel. Díky návrhu architektury do čtyř vrstev by změna vstupní tabulky ve vrstvě 00 neměla ovlivnit výsledná data za předpokladu, že ve vrstvě 20 jsou tyto rozdíly ve struktuře zohledněny.

Navrhované řešení splňuje stanovený hlavní cíl práce, a to implementaci datového skladu jakožto automatizovaný nástroj pro centrální uchovávání dat jako náhradu za stávající proces sestávající se převážně z ruční úpravy denně aktualizovaných dat.

6 Seznam použité literatury

1. **Goldmeier, Jordan a Gutman, Alex J.** *Becoming a Data Head: How to Think, Speak, and Understand Data Science, Statistics, and Machine Learning.* New Jersey : John Wiley & Sons, 2021. 9781119741749.

2. **Békés, Gábor a Kézdi, Gábor.** *Data Analysis for Business, Economics, and Policy.* Cambridge : Cambridge University Press, 2021. 9781108483018.

3. **Deepak, Gupta, Siddhartha, Bhattacharyya a Ashish, Khanna.** *Intelligent Data Analysis: From Data Gathering to Data Comprehension.* New Jersey : John Wiley & Sons Inc, 2020. 9781119544456.

4. **Aggarwal, Charu C.** *Data Mining The Textbook.* New York : Springer, 2015. 978-3319141411.

5. **Extension, UC Berkeley.** Berkeley Extension. *Understanding Data Science Roles: Who Does What?* [Online] [Citace: 9. 1 2023.] https://bootcamp.berkeley.edu/blog/understanding-data-science-roles-who-does-what/.

6. *The CRISP-DM Model: The New Blueprint for Data Mining.* **Shearer, Colin.** 4, Washington : THE DATA WAREHOUSING INSTITUTE, 2000, Sv. 5.

7. **Microsoft.** Microsoft Learn. *What is the Team Data Science Process?* [Online] Microsoft. [Citace: 10. 1 2023.] https://learn.microsoft.com/en-us/azure/architecture/data-science-process/overview.

8. **Rainardi, Vincent.** *Building a Data Warehouse With Examples in SQL Server.* London : Apress, 2014. 1430211962.

9. **MIcrosoft.** Microsoft Learn. *Microsoft Learn.* [Online] Mircrosoft. [Citace: 1. 2 2023.] https://learn.microsoft.com/en-us/office/troubleshoot/access/database-normalization-description.

10. **Sinha, Tammay.** IBM Cloud Blog. *IBM Cloud Blog.* [Online] IBM, 2021. 5 16. [Citace: 14. 1 2023.] https://www.ibm.com/cloud/blog/olap-vs-oltp.

11. **Kimball, Ralph a Ross, Margy.** *The Data Warehouse Toolkit, 3rd Edition: The Definitive Guide to Dimensional Modeling.* místo neznámé : Wiley, 2013. 9781118530801.

12. **Královéhradecký kraj.** Data KHK. *Data KHK.* [Online] 12. 3 2023. [Citace: 12. 3 2023.] https://www.datakhk.cz/.

13. **Národní katalog otevřených dat.** Portál otevřených dat. *Portál otevřených dat.* [Online] [Citace: 1. 1 2023.] https://data.gov.cz/datov%C3%A1-sada?iri=https%3A%2F%2Fdata.gov.cz%2Fzdroj%2Fdatov%C3%A9-sady%2F00024341%2F2f68f5591244f17225bae7270d79c589.

14. **Katalog otevřených dat.** Katalog otevřených dat. *Katalog otevřených dat.* [Online] [Citace: 1. 1 2023.] https://opendata.mzcr.cz/dataset/ciselnik-orp.

15. **Reis, Joe a Housley, Matt.** *Fundamentals of Data Engineering: Plan and Build Robust Data Systems.* Sebastopol : O'Reilly Media Inc., 2022. 9781098108304.

16. **Han, Jiawei, Kamber, Micheline a Pei, Jian.** *Data Mining: Concepts and Techniques.* Burlington : Morgan Kaufmann , 2005. 9781558609013.

17. **Adamson, Jeremy.** *Minding the Machines: Building and Leading Data Science and Analytics Teams.* New York : Wiley, 2021. 1119785324.

7 Přílohy

1) Veškeré zdrojové kódy dostupné na URL adrese https://github.com/martingold/datakhk-data-warehouse

Univerzita Hradec Králové
Fakulta informatiky a managementu

Zadání diplomové práce

Autor: **Bc. Martin Gold**

Studium: I2100845

Studijní program: N0688A140019 Datová věda

Studijní obor: Datová věda

Název diplomové práce: **Analýza a optimalizace procesu zpracování otevřených dat Královehradeckého kraje**

Název diplomové práce AJ: Analysis and optimization of the data mining process of the Hradec Králové Region

Cíl, metody, literatura, předpoklady:

Cílem práce je analyýza stávající řešení zpracování otevřených dat Královéhradeckého kraje a návrh řešení pro efektivnější a automatizované fungování.

1. Úvod

2. Datový portál Královéhradeckého kraje

3. Analýza stávajícího řešení

4. Implementace automatizovaného řešení

5. Možné budoucí využití vlastního datového skladu

1. Reis, Joe a Housley, Matt. Fundamentals of Data Engineering: Plan and Build Robust Data Systems. Sebastopol : O'Reilly Media Inc., 2022. 9781098108304.

2. Rainardi, Vincent. Building a Data Warehouse With Examples in SQL Server. London : Apress, 2014. 1430211962.

3. Aggarwal, Charu C. Data Mining The Textbook. New York : Springer, 2015. 978-3319141411.

4. Goldmeier, Jordan a Gutman, Alex J. Becoming a Data Head: How to Think, Speak, and Understand Data Science, Statistics, and Machine Learning. New Jersey : John Wiley & Sons, 2021. 9781119741749.

5. Han, Jiawei, Kamber, Micheline a Pei, Jian. Data Mining: Concepts and Techniques. Burlington : Morgan Kaufmann , 2005. 9781558609013.

Zadávající pracoviště: Katedra informatiky a kvantitativních metod, Fakulta informatiky a managementu

Vedoucí práce: doc. RNDr. Petra Poulová, Ph.D.

Datum zadání závěrečné práce: 26.1.2021