Západočeská univerzita v Plzni
Fakulta aplikovaných věd
Katedra informatiky a výpočetní techniky

Bakalářská práce

Datová kostka pro analýzy výzkumu a vývoje inovací pro datový sklad ZČU

Plzeň 2022 Dominik Jež

ZÁPADOČESKÁ UNIVERZITA V PLZNI
Fakulta aplikovaných věd
Akademický rok: 2021/2022

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

(projektu, uměleckého díla, uměleckého výkonu)

Jméno a příjmení:	**Dominik JEŽ**
Osobní číslo:	**A19B0078P**
Studijní program:	**B0613A140015 Informatika a výpočetní technika**
Specializace:	**Informatika**
Téma práce:	**Datová kostka pro analýzy výzkumu a vývoje inovací pro datový sklad ZČU**
Zadávající katedra:	**Katedra informatiky a výpočetní techniky**

Zásady pro vypracování

1. Seznamte se s teorií datových skladů a strukturou registru CEP (centrální evidence projektů).
2. Navrhněte model datové kostky pro porovnání výstupů realizovaných projektů českých vysokých škol, ústavů a veřejných výzkumných institucí. Navrhněte datovou pumpu pro výběr relevantních dat z CEPu.
3. Realizujte datovou kostku a datovou pumpu v dostupném programovém vybavení pro systému řízení báze dat Oracle.
4. Vytvořenou datovou kostku nasaďte do testovacího prostředí datového skladu ZČU. Ověřte funkčnost datové pumpy naplněním dat do připravené datové kostky.
5. Vhodným analytickým nástrojem pro datové sklady vyzkoušejte reprezentativní analytické dotazy nad vytvořenou datovou kostkou.

Rozsah bakalářské práce: **doporuč. 30 s. původního textu**
Rozsah grafických prací: **dle potřeby**
Forma zpracování bakalářské práce: **tištěná/elektronická**

Seznam doporučené literatury:

Dodá vedoucí bakalářské práce.

Vedoucí bakalářské práce: **Ing. Lenka Jirsová**
Centrum informatizace a výpočetní techniky

Konzultant bakalářské práce: **Ing. Martin Zíma, Ph.D.**
Katedra informatiky a výpočetní techniky

Datum zadání bakalářské práce: **4. října 2021**
Termín odevzdání bakalářské práce: **5. května 2022**

L.S.

Doc. Ing. Miloš Železný, Ph.D.
děkan

Doc. Ing. Přemysl Brada, MSc., Ph.D.
vedoucí katedry

V Plzni dne 14. října 2021

Prohlášení

Prohlašuji, že jsem bakalářskou práci vypracoval samostatně a výhradně s použitím citovaných pramenů.

V Plzni dne 2. května 2022

Dominik Jež

Poděkování

Tímto bych rád poděkoval své vedoucí bakalářské práce Ing. Lence Jirsové za odborné rady a pomoc při zpracování této práce. Dále bych chtěl poděkovat konzultantovi bakalářské práce Ing. Martinu Zímovi, Ph.D. za věcné připomínky.

Abstract

This presented bachelor's thesis deals with the issue of data warehousing and the process of data gain from the CEP, which belongs to the International System of Research, Development, and Innovations.

This thesis describes the process of retrieving data from the API IS VaVal in the JSON format, the recording of the data to DBMS Oracle and finally the verification of the data's relevance and its cleansing.

The second part of the thesis presents a model of a data dart, which is filled with the previously mentioned data. The precision of the data's function is proved by running a few analyses in the Power Bi software. The correctness of the data is confirmed in the DBMS after uploading the unedited API data.

Key words

Data warehouse, Central Evidation of Projects, Data cube, Databsase, Data transformation

Abstrakt

Bakalářská práce se zabývá problematikou datových skladů a získáváním dat z CEP, který spadá pod Informační systém výzkumu, vývoje a inovací.

V práci je popsán proces získávání dat z API IS VaVaI ve formátu JSON, nahrání dat do SŘBD Oracle, zjišťování relevance dat a jejich čistění.

Ve druhé části je vytvořen model datové kostky, který je naplněný daty. Ověření správnosti funkčnosti probíhá pomocí jednoduchých analýz v nástroji Power BI. Správnost dat je otestována v SŘBD z neupravených nahraných dat API.

Klíčová slova

Datový sklad, Centrální Evidence Projektů, Datová kostka, Databáze, Transformace dat

Obsah

Seznam obrázků

Seznam tabulek

Seznam kódů

1 Úvod

Centrum informatizace a výpočetní techniky poskytuje služby Západočeské univerzitě v Plzni, které jsou spojené s rozvojem a provozem informačních technologií. Jedno odvětví centra se věnuje tvoření datových skladů, pomocí kterých lze analyzovat data sloužící např. k výkaznictví výročních zpráv, k tvorbě a udržení strategických záměrů univerzity, dalším analýzám či hledání nečekaných korelací. Zároveň datový sklad vede díky svým principům (jen čte, neopravuje data) k čistění dat v primárních systémech.

V dnešní době je problém s velkým množství dat. V práci jsou vybrána vhodná data z CEPu (Centrální Evidence Projektů), která budou sloužit pro naplnění datové kostky. Vybraná data z datové kostky nám slouží pro vizualizaci ve vhodném analytickém nástroji, kde můžeme narazit na zajímavé korelace dat.

Nejdříve bude rozebrána teoretická část databázových systémů a databázových skladů kvůli seznámení a objasnění pojmů. Následovat bude popis struktury registru CEP. Ve druhé části práce bude popsáno stažení dat CEP z Informačního systému výzkumu, vývoje a inovací (IS VaVaI). Po stažení dat bude následovat nahrání a transformace dat do analytické databáze Oracle, kde budou data podrobena analýze kvůli návrhu a tvoření datové kostky. Nahrávání dat do datového skladu bude obstarávat datová pumpa.

Pro ověření správnosti implementace bude nasazena datová kostka do testovacího prostředí datového skladu Západočeské univerzity. Vyhodnocení výsledků datové kostky proběhne v analytickém nástroji Power BI[1], kde jsou výsledky prezentovány v pomocí dashboardů. Otestování integrity výsledků z grafů bude provedeno v databázi, na datech, která jsou stažena z IS VaVaI a nebyla nijak editována.

[1]Business Intelligence – označení pro analytické a vykazovací činnosti, které slouží jako podklady pro rozhodování.

2 Databáze

Databáze je logická množina souvisejících dat uložených pohromadě, tak aby splnila informační potřeby systému. Pro jednoduchost budeme předpokládat, že pojmy *databáze* (DB) a *databázový systém* (DBS) budou ekvivalentní.

2.1 Systém řízení báze dat

Jedná se o software pro správu a údržbu kolekcí dat. Data lze v **systému řízení báze dat** (SŘBD) definovat, vkládat, manipulovat, mazat a obnovovat. Obsahuje programy, které se starají o správný chod databáze, tzn. tvoří rozhraní mezi aplikacemi a uloženými daty. Aby dané softwarové vybavení mohlo být považované za SŘBD, musí umět pracovat s velkým množstvím dat a zároveň mít definovanou strukturu. SŘBD pracuje s transakcemi, aby zachoval konzistentní stav. Operace během transakcí musí být atomické – např. výběr peněz z banky je zajištěn dvěma kroky – první krok je odečtení peněz z účtu a druhý krok je vydání peněz. Tyto dva zmíněně kroky musí proběhnout celé. V případě neúspěchu se databáze dostane do nekonzistentního stavu. Proto je dobré sdružovat tyto operace do transakcí, které mají vlastnosti ACID [8], kde každé písmeno označuje klíčový bod. Následující zkratky budou vypsané v angličtině a popsané v češtině:

- **A**tomicity (atomicita) – transakce se musí provést celá. Jestliže se nepovede vše úspěšně, transakce bude vrácena do původního stavu.
- **C**onsistency (konzistence) – po vykonání transakce nebude narušena konzistence. Znamená to, že databáze zůstane konzistentní.
- **I**solation (izolace) – data, která se aktuálně využívají v transakcí, nesmí být použita jinou transakci. Transakce se nesmí navzájem ovlivnit. Možné zavinění může vést do nekonzistentního stavu.
- **D**urability (trvalost) – jestliže je transakce dokončena úspěšné, tak je zaznamenaná trvale a v případně nedostupnosti se o data nepřijde.

2.2 Databázový systém

Pro zjednodušení si lze představit databázi jako kartotéku v lékařské ordinaci, kde záznamy o pacientech představují **data**, která jsou uložená na **záznamovém médiu**. Aby, bylo možné se vyznat, v uložených záznamech, je zapotřebí vhodné **členění** seřazené podle abecedy. Jestliže je nutné upravit informace o pacientovi nebo přidat nového pacienta, tak se musí použít příslušné **nástroje**, které to umožňují. Problém nastává, když je dat přespříliš a správa dat se stává neúnosná. V tuto chvíli je nutné záznam uložit na médium, který je zpracováván systémem. [36]

- DBS je navržen tak, aby byl umožněn přístup k datům všem uživatelům ve stejnou dobu,
- DBS se snaží zabránit duplicitám,
- DBS umožňuje flexibilní přístup. To znamená, že může více uživatelů vytvářet dotazy, ale modifikovat data můžou jen ti, kteří mají vyhrazená práva.

Databázový systém je logické seskupení dat. Obsahuje relevantní data, která jsou uložená v tabulkách. Tabulky se dále člení do řádků a sloupců, kde sloupce obsahují názvy atributů.

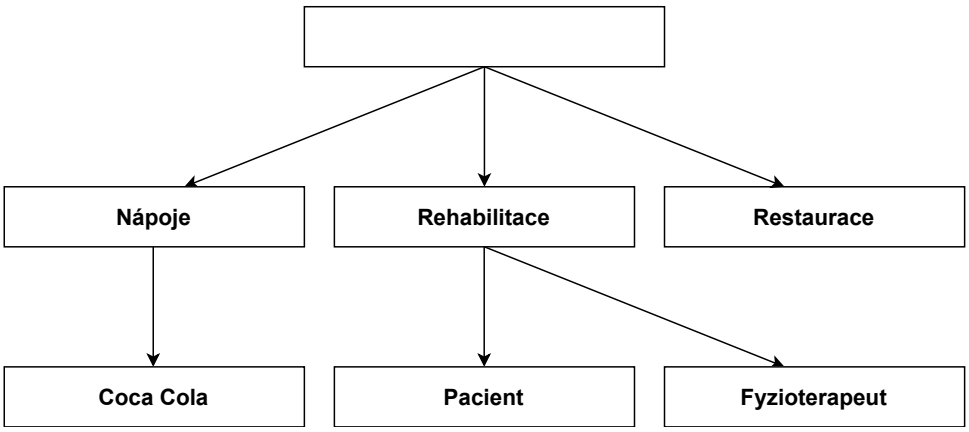
2.2.1 Logické databázové modely

S vývojem technologií se přišlo na více způsobů, jak ukládat a zpracovávat data. Nejedná se o samostatný vývoj modelu, ale i způsob zpracování dat v SŘBD. Zde bude následovat stručný popis databázových modelů seřazených od nejstaršího po nejnovější. Nejvíce používaný je nyní relační, kde byla snaha vytlačit ho objektovým modelem [15, 19]. To se však nepovedlo. Na úkor toho se začala používat objektově relační databáze (ORDB), která nese výhody jak relační, tak objektové databáze [3].

Hierarchická databáze

Tento model byl vyvinut v 60. letech. Data jsou uložená do stromu (obrázek 2.1). Stromem je souvislý graf, který neobsahuje kružnice. Strom podporuje rychlé vyhledávání. Podporuje se názvosloví jako je kořen, vnitřní uzel, list, rodič, potomek, apod. Pro vyhledávání, kde kořen nepatří nikomu a na něj jsou nabalované záznamy, které už můžou obsahovat spojitost mezi dalšími záznamy [5]. Problém nastává v případě, kdy lze záznam zařadit pod

vícero rodičů (nebo je to nutností) – to ve stromové struktuře nelze provést efektivně. Proto vznikl nový druh databázového modelu – síťový model.



Obrázek 2.1: Hierarchický model (zdroj vlastní)

Síťová databáze

Navazuje plynule na hierarchický model, kde je přidána vazba, že entita může mít více rodičů. Byla přidána rekurze, kde entita může ukazovat sama na sebe. Na druhou stranu s rozšířením síťového modelu nastaly problémy ohledně přehlednosti a pružnosti databáze [5]. Používání toho modelu v průběhu 70. let vystřídal model relační.

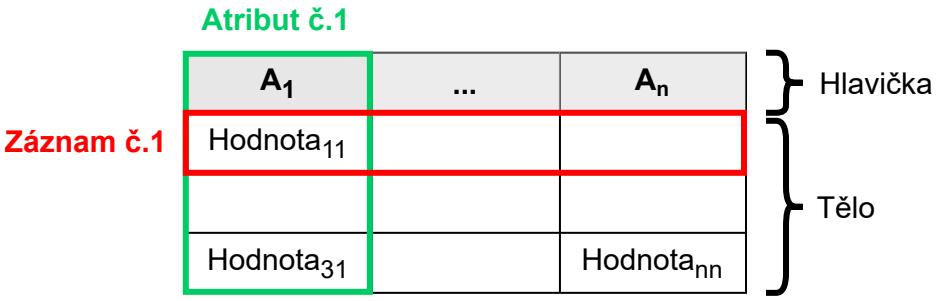
Relační databáze

Relační databáze (RDB) vnikly z potřeby vytvoření modelu, kde pomocí matematických operací (kartézský součin, selekce, projekce, rozdíl, sjednocení a spojeni/přejmenování) lze manipulovat s daty – matematické operace lze nazvat jako **operátory**. Pomocí těchto základních matematických operací lze uskutečnit i složitější konstrukce. Pro práci s relacemi lze použít jeden nebo více operátorů, kde vstupem je jedna nebo více relací a výstupem je jedna výstupní relace. Tento model byl představen pánem E. F. Codd v roce 1970 [9].

RDB používá **Systém Řízení Báze pro Relační Databáze**[1] (RDBMS) a skládá se z tabulek (relací), které se dají rozdělit na **atributy** a **záznamy** – atributy označují sloupce, záznamy řádky. Samotná tabulka se dá rozdělit na dvě části: hlavička a tělo. Hlavička obsahuje názvy atributů a tělo samostatné záznamy (řádky) – obrázek 2.2. Pro úplnost – atributy jsou určené

[1]Relational Database Management System

datovým typem a doménou [19]. Datový typ jako v ostatních programovacích jazycích může nabývat libovolných hodnot jako např. *Integer*, *String*, *Double*, *Bollean* či dalších. Doména určuje, jakých hodnot může atribut nabývat – např. známky ve škole.



Obrázek 2.2: Relační tabulka (zdroj vlastní)

Začátkem 70. let byl vytvořen jazyk SEQUEL (později zkrácen jako **SQL**) ve firmě IBM (International Business Machines Corporation) pány Donald D. Chamberlin a Raymond F. Boyce, který slouží jako strukturovaný dotazovací jazyk pro práci s RDB.

Zástupci: Oracle Database, Microsoft SQL Server, MYSQL, MS Access.

Objektová databáze

U objektové DB jsou veškerá data zaznamenávaná jako objekty – více se podobá objektům z reálného světa. Platí zde principy objektově orientovaného programování jako zapouzdření, dědění či polymorfismus. Samotné objekty jsou uložené v databázích jako kolekce (Array, List, Set, apod.) [16, 34]. Je hlavním konkurentem relačních databází.

Zástupci: Wakanda, ObjectStore.

Objektově relační databáze

ORDB je hybrid mezi relačním a objektovým modelem.

2.2.2 Normální formy

Normalizace slouží k přeuspořádání databáze do takové podoby, aby obsahovala co nejryzejší podobu – tedy žádné duplikace a korektní uložení dat. Každá tabulka je při dodržování forem vhodně rozložena nebo by měla být

zkontrolována, zda splňuje podmínky 1-5 normální formy (NF) + Boyce—Coddova normální forma (BCNF). Při splnění všech požadavků se tabulka může označit za normalizovanou. V praxi se DBS snaží splnit podmínky alespoň 3. NF, kde splnění 3. NF podmiňuje splnění 1. a 2. NF. Při normalizování databáze lze efektivně vyhledávat, třídit, ukládat a lze přijít na nepříznivé anomálie. Pro různé stupně strukturovaní se dělí NF do vícero částí, které budou popsány od 1. NF do 5. NF včetně BCNF, protože patří mezi nejpoužívanější.

1. NF

Relace se nachází v 1. NF jestliže atributy jsou dále nedělitelné. Příkladem může být celá směrovací adresa zaznamenána v jednom atributu, která se dá rozdělit na separátní atributy jako město, ulice, č. popisné a poštovní adresa.

2. NF

Musí se nacházet relace v 1. NF a zároveň každý neklíčový atribut musí být závislý na celém klíči a nikoli pouze na jeho části.

3. NF

Musí se nacházet relace v 2. NF a zároveň všechny neklíčové atributy nesmí být tranzitivně závislé na klíči. Jinak řešeno, nesmí existovat závislost mezi neklíčovými atributy.

Boyce–Coddova normální forma

Opět se předpokládá, že pro splnění BCNF musí být relace v 3. NF a zároveň v každé funkční závislosti je její determinant kandidátem klíče (neprimární atributy můžou být závislé na primárním klíči, ale i na neprimárním (kandidátním) klíči).

Relace, která splňuje BCNF, splňuje zároveň podmínky 3. NF. Nemusí ale platit tato skutečnost obráceně, když platí 3. NF, tak platí BCNF, protože může existovat atribut, který je společný pro kandidátní klíče, které můžou reprezentovat primární klíč.

4. NF

Relace musí splňovat podmínky BCNF a nesmí obsahovat sloupce, které mezi sebou nemají žádnou relaci – tzn. pro relaci A –> B, kde pro hodnotu

A existuje více hodnot B, tak se považuje za mnohohodnotovou závislost (multi-valued dependency) a je potřeba pro hodnoty B vytvořit samostatnou tabulku, která bude odkazovat do hlavní tabulky. [12]

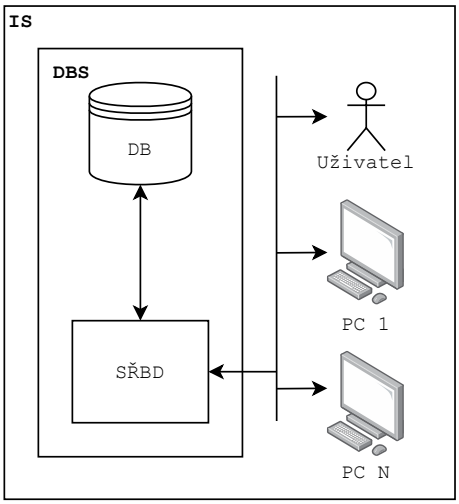
5. NF

5. NF se taky označuje jako PJNF (project-join normal form). Relace musí splňovat podmínky 4. NF a zároveň nesmí obsahovat žádné závislosti – tedy veškeré snahy o propojení tabulek pomocí JOIN není umožněno. Snahou je hlavní tabulku rozbít, na co nejvíce malých tabulek, kde je zapotřebí se vyhnout redundancím. V praxi se 4. NF a 5. NF příliš nepoužívá kvůli obtížnějšímu dotazování a nepřehlednosti. [13]

2.3 Informační systém

Přesná definice pojmu Informačního Systému (IS) neexistuje – to z důvodu, že každý uživatel považuje IS za trošku něco jiného, ať už z důvodu terminologie nebo zdůraznění dané problematiky.

Lze ji popsat jako trojici: proces, data a prostředí. Procesem je myšlena práce s daty, jak už samotné přidání, odebrání, uložení, distribuce či transformace. Data tvoří samostatný informační celek, které slouží jako hlavní zdroj. Prostředím je myšleno vše okolo procesů a dat. Tvoří je vše, co dokáže ovlivnit DBS – lidé, objekty nebo vlastnosti (rozhodování). Shrnutí, jak si lze Informační systém představit, popisuje obrázek 2.3, DB představují data, SŘBD procesy a prostředí uživatelé.



Obrázek 2.3: Prvky Informačního systému (zdroj vlastní)

3 Datové sklady

Datový sklad (data warehouse, DW) je velká kolekce organizovaných dat relevantních pro manažerské rozhodování a další analýzy. Data pochází z mnoha různých zdrojů (RDB, surová data), které se musí pročistit a transformovat do uceleného zdroje pro analytické účely. Tomu nám pomůžou nástroje pro **ETL** (Extract, Transform and Load). Pouze takto upravená data můžou sloužit pro řízení vztahu se zákazníky (Customer Relationship Management, CRM) nebo pro manažerské účely.

Pro provedení ETL přichází na řadu práce datových analytiků, datových vědců a datových inženýrů, kteří mají na starost prohledávání informací, údržbu dat a hledání korelací v datech. K tomu lze využít různých nástrojů: strojové učení a další podpůrné programy.

3.1 Čerpání dat

Jak již bylo popsáno, data, která lze využít, pocházejí z různorodých zdrojů. Primární zdroje dat, které se bezesporu nejméně upravují, se nachází v interních DB. Dalším zdrojem můžou být různé aplikace REST (Representational state transfer) či SOAP (Simple object access protocol) služby[1], kde lze po domluvě nebo když jsou volně přístupné, použít a natáhnout data do příslušného vybavení. Hodně informací se rovněž může nacházet na internetových stránkách, kde pomocí extrahovaných nástrojů, lze opět tyto data získat.

3.1.1 Datová pumpa

Častěji označováno jako spojení tří písmen **ETL** resp. **ELT** (Extract, Load and Transform) [32]. Klasickým problémem je slučování dat z různých zdrojů, protože obsahují odlišnou strukturu dat. Cílem procesu je data očistit a nahrát do jasně definovaného struktury (obrázek 3.1).

První písmeno značí **Extrahování**. Během extrahování různých zdrojů se nahrají data na jednu větší hromadu – kupříkladu do datového jezera (častěji používáno „Data Lake“)[2] nebo do prozativní DB (staging DB). Tento

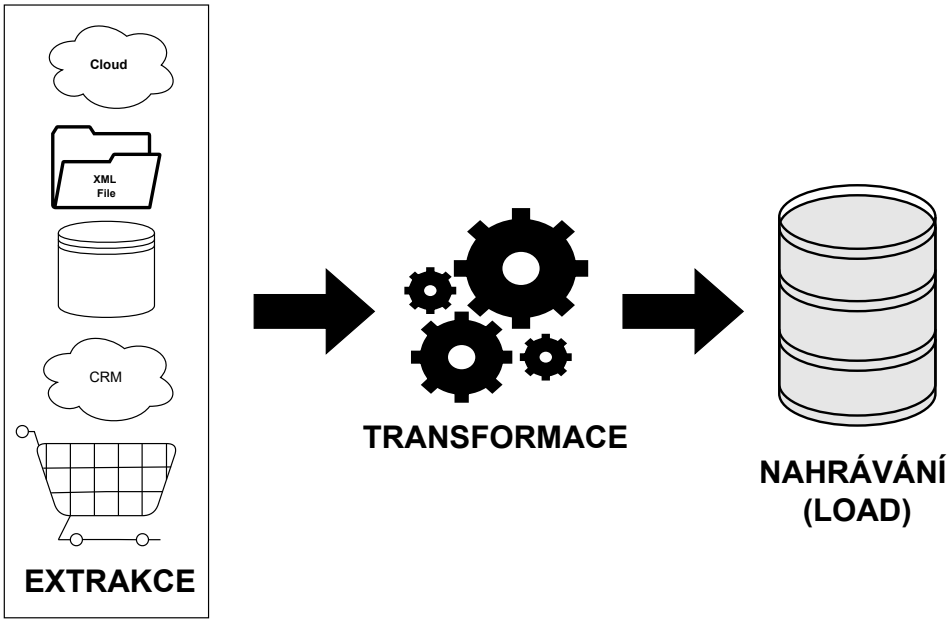
[1] `https://www.redhat.com/en/topics/integration/whats-the-difference-between-soap-rest`

[2] Místo, kde je spoustu nestrukturovaných dat.

postup může být proveden pomocí komerčních i placených integračních nástrojů či ručně.

Druhé písmeno značí **Transformaci**. V téhle fázi jsou data nahrána v prostředí, kde probíhají operace jako čistění dat, odstraňování duplicit, filtrování, agregace nebo také ověřování dat. Cílem je data standardizovat.

Poslední písmeno značí nahrání, v angličtině **Load**. Zde se data nahrají na nové místo již v jasně definované struktuře – buď se nahrají celá a nebo po částech z důvodu velké velikosti.



Obrázek 3.1: ETL (zdroj vlastní)

Často jsou tyto operace spouštěné paralelně kvůli úspoře času, protože během transformace dat, která zabere čas, se můžou další data extrahovat.

Zástupci ETL nástrojů: Informatica, Talend, Pentaho Kettle, apod. [2].

3.2 Datové tržiště

Datové tržiště (anglicky Data mart) je součástí DW – podmnožina/konkrétní část, např. data pro finanční oddělení. Vytváří se v případě, kdy data v DW jsou nepřehledná nebo je jich přespříliš a znemožňují práci se samostatným celkem. Datové tržiště je určeno především pro business, aby mohl ovlivnit rychleji obchodní procesy – možnost modifikace dat (není nutnost mít možnost data za posledních 10 let, ale je dostačující poslední měsíc). Většinou

se tvoří více datových tržišť pro oddělené divize. Pro lepší porovnání datové tržiště a DW slouží tabulka 3.1. [31]

Vlastnosti	Datový trh	Datový sklad
Velikost	< 100 GB	100 GB +
Rozsah	Jedno oddělení	Celý podnik
Integrace dat	Jedna oblast	Všechny oblasti
Čas k reálnému provozu	Minuty, týdny, měsíce	Měsíce až roky
Zdroje dat	Málo/Několik	Obsáhlé

Tabulka 3.1: Porovnání DW a datového trhu [31]

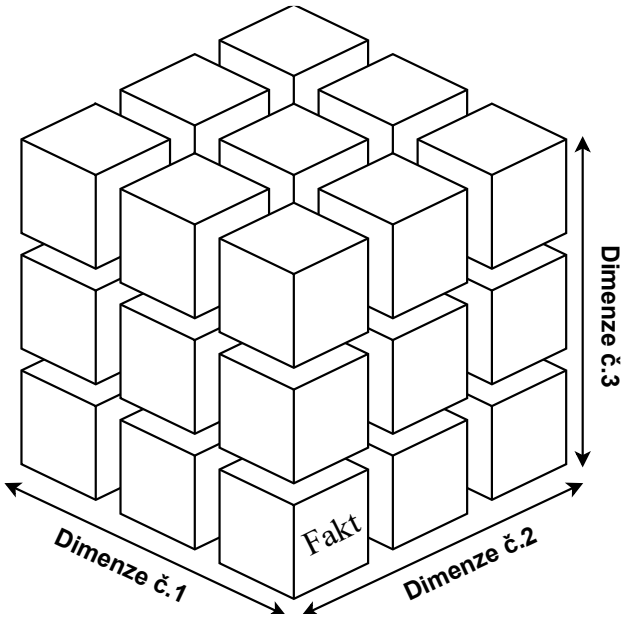
3.3 Datová kostka

Datová kostka je multidimenzionální model – většinou se skládá ze 3 a více tabulek, kde každá tabulka reprezentuje dimenzi či fakt. Důvodem pro vytváření datové kostky je snaha zobrazit abstrakci nad daty z různých hledisek. Používá se tam, kde je potřeba zobrazit data pohromadě, kde je nutné ukázat souvislost. Např. máme tří-dimenzionální kostku, kde 1. dimenze představuje výrobky, 2. dimenze představuje státy a 3. dimenze představuje datum v letech. 4. tabulka modelu obsahuje faktové (součtové) hodnoty odpovídající těmto dimenzím. Pro lepší představení je zde obrázek 3.2. Nejvíce používanými schématy je hvězda (star) a sněhová vločka (snowflake) [33].

3.3.1 Schéma hvězda

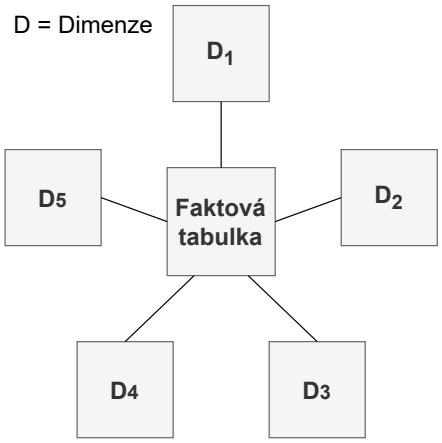
Kvůli své přehlednosti a jednoduchosti je hvězda nejpoužívanější schéma u DW. Hvězda obsahuje faktovou tabulku a 1 nebo více dimenzí (obrázek 3.3). **Tabulka faktů** může obsahovat cizí klíče (Foreign Key, FK), které ukazují na tabulky dimenzí nebo jestliže jsou atributy velice prosté, pak jsou považována za fakta samotná (např. pohlaví). **Tabulka dimenze** obsahuje primární klíč a statická data. Dimenzionální tabulky můžou obsahovat spousty dat – v řádech od tisíců až po miliony a více. Nevýhoda schématu hvězdy je častá redundance dat. Na druhou stranu poskytují velice slušný výkon a jsou snadné jak na pochopení, tak i vytvoření. DW nesplňuje normální formy jako OLTP, protože se s tím lépe pracuje.

Kupříkladu můžeme mít faktovou tabulku, která obsahuje 3 dimenze (3 cizí klíče) a vlastní fakta jako pohlaví a výše příjmů. Dimenze č.1 obsahuje datum, který se člení na dny, měsíce a roky. Dimenze č.2 obsahuje jméno pro-



Obrázek 3.2: Datová kostka (zdroj vlastní)

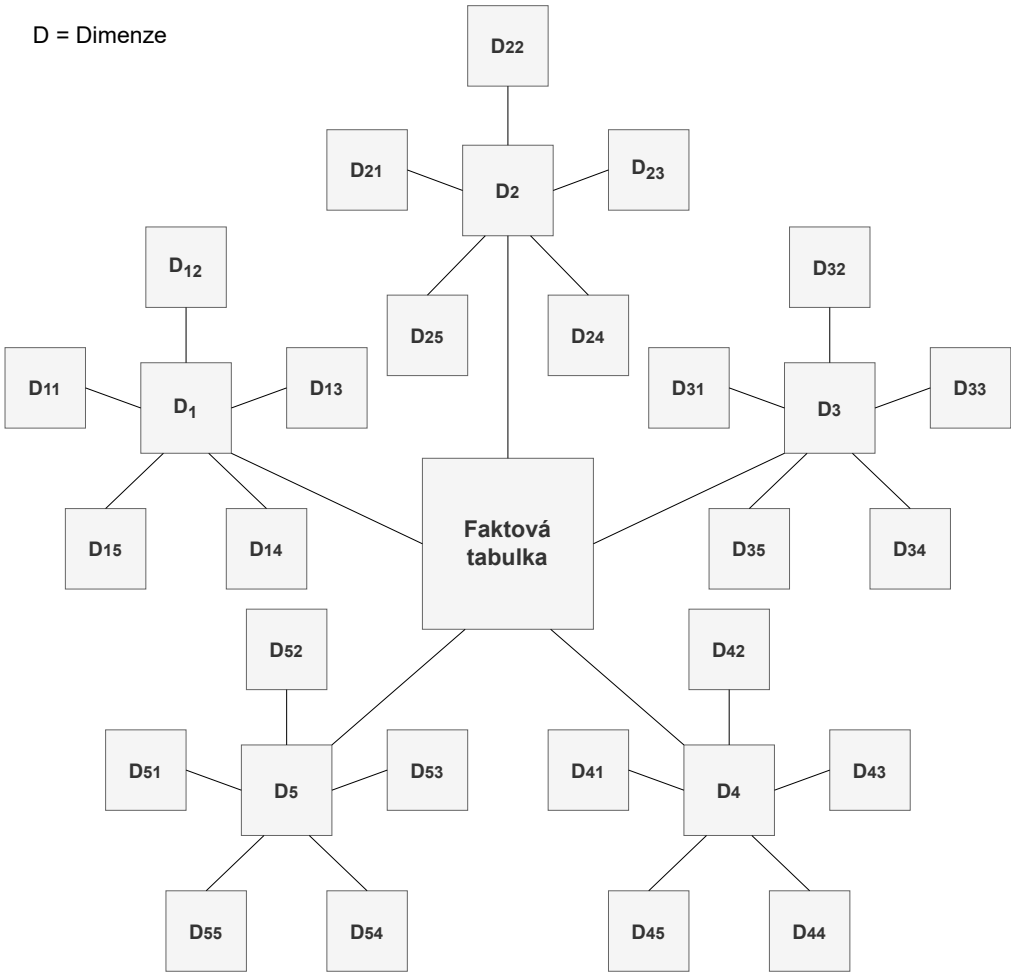
dejce, které je rozděleno na identifikační údaje jako je: jméno, příjmení a občanský průkaz. Dimenze č.3 obsahuje auto, který lze rozpoznat podle značky, výkon, velikost, barva a model. Cílem bude zjistit, kdo měl vyšší příjem více než milion korun v roce X, za prodaná auta Y a bude vypsán jako Z [33].



Obrázek 3.3: Schéma hvězda (zdroj vlastní)

3.3.2 Schéma sněhová vločka

Má hodně blízko hvězdicovému schématu, kde je zásadní rozdíl v normalizování tabulek dimenzí. Každé schéma obsahuje tabulku faktů a opět jako předešlého schématu 1 nebo více dimenzí. Navíc tyto dimenze můžou obsahovat další dimenze (jsou tedy zařazené hierarchicky). Snahou je se úplně nebo částečně vyhnout duplicitám, a tím zpřehlednit tabulky za cenu výkonu. Schéma sněhové vločky je o něco více složitější než schéma hvězdy. Pro lepší pochopení členění dimenze slouží obrázek 3.4, kde si dimenzi D1 lze představit jako značku automobilu a pod-dimenze D11, D12 a D13 jako model, barvu a výkon motoru.



Obrázek 3.4: Schéma sněhová vločka (zdroj vlastní)

3.4 OLTP

Jedná se o způsob, jakým jsou data v databázi uložena. U OLTP (Online Transaction Processing) je cílem umožnit, co nejsnadnější modifikaci prostředí – jsou zachovány operace jako výběr (`SELECT`), vkládání (`INSERT`), aktualizování (`UPDATE`) či mazání (`DELETE`) záznamů.

Tento způsob se praktikoval u všech databází než se postupně začalo využívat OLAP (Online Analytical Processing). Ten je vhodnější tam, kde je nutná interakce s databázemi – webové stránky, mobilní aplikace, podniky a další [24]. Nutno podotknout, že se jedná opět o vlastnosti **ACID**.

3.5 OLAP

OLAP se využívá v případech, kde je cílem uspořádat velké množství dat tak, aby sloužila k analytickým účelům (použití v datových skladech/datových trzích). Prvotní zpracování dat v OLAP je pomalejší než v OLTP. Z toho důvodu, že se přidává hojně indexace dat, aby se naopak urychlilo dotazování. Nepředpokládá se, že data budou modifikována, proto jsou zde běžně odebrána oprávnění: `INSERT`, `UPDATE` a `DELETE` – jedná se v podstatě o protiklad OLTP.

Zároveň se hojně používá s technologiemi jako: **data mining**, **business intelligence**, **reportovací nástroje** nebo se spojitostí s předpovědí/prognózou [28]. Hlavní důvod, proč využívat OLAP je podpora tvoření datové kostky (**OLAP cube**). Se spojitostí s faktovou tabulkou a dimenzemi tvoří snadné a rychlé (z pohledu uživatele) dotazování nad velkým množstvím dat.

Pro úplnost je zde uvedená tabulka 3.2, která shrnuje základní rozdíly mezi **OLTP** a **OLAP**:

3.6 Data Mining

Česky ekvivalent *Dolování dat* není hojně rozšířen (bude se používat anglická terminologie). Pojmy ETL a Data Mining spolu úzce souvisí, protože ETL je část konstrukce v Data Mining-u. Z opačného hlediska data Mining úzce souvisí s tzv. Data Science[3] a lze ho pod něj zařadit.

Data Mining je proces, pomocí kterého se hledají vybraná data ve velkých

[3]Český název: Datová věda – využívá vědecké procesy pro získávání znalostí z dat

	OLTP	OLAP
Orientace DB	transakční	vyhodnocovací (analytická)
Uživatel	úředník, DB admin.	manažer, analytik
Návrh DB	aplikačně orientovaný	věcně orientovaný
Sumarizace dat	základní, vysoce detailní	shrnutá, kompaktní
Přístupy	číst, psát, modifikovat	pouze číst
Zaměření	vkládání dat	získávání informací
Počty záznamů	desítky	miliony
Přednosti	vysoký výkon/propustnost	vysoká flexibilita
Odezva dotazu	pomalá	rychlá

Tabulka 3.2: OLTP vs. OLAP [30]

objemech dat. Dá se dělit na dvě hlavní části – k popisu datasetu nebo slouží spolu se strojovým učením k předpovědi výsledků [4] – např. oslovení skupiny zákazníků v obchodě. Zároveň v částech je obsaženo detekování a filtrování dat.

Občas se naskytne termín **KDD** (Knowledge Discovery in Databases[4]), který se bere jako synonymum ke Data Mining-u. Rozdíl spočívá pouze v tom, že KDD obsahuje přípravu samostatných dat. Dá se považovat Data Mining za součást KDD [11].

Celý proces Data Mining-u lze rozdělit do 6 kroků [26]:

1. **Definice problému** – návrh a tvoření plánu (porozumění požadavků a stanovení cíle).
2. **Porozumění datům** – tvoření prvních hypotéz (během procesu potvrdit, vyvrátit nebo najít nové hypotézy).
3. **Příprava dat** – integrace, čistění a úprava dat do nutné podoby kvůli business inteligence nástrojům; nutno dát důsledný pozor na integraci dat, jinak vede ke znehodnocení celého řešení.
4. **Modelování** – testování metod za cenu získání nejlepších výsledků.
5. **Hodnocení** – zvážit konečné zhodnocení ze získaných modelů a zvážit možnost nasazení.
6. **Nasazení** – nasazení do ostrého prostředí; nutné je dbát na aktuálnost dat (potřeba se starat o data a udržovat je aktuální).

[4]Český název: Dobývání znalostí z databází

3.6.1 Používané techniky

Během používání procesu Data Mining-u vzniklo několik způsobů, jak se k datům dostat. Každý způsob může obsahovat o něco jiné výsledky z důvodu použitých technik. Bude následovat kratší výčet metod, které se využívají: [11]

- **Regresní metody** – ne/lineární regresní analýza, neuronové sítě (NS).
- **Klasifikace** – diskriminační analýza, rozhodovací stromy, NS.
- **Segmentace (shlukování)** – genetické algoritmy, Kohoneovy mapy.
- **Analýza vztahů** – odvozování pravidel (if A then B).
- **Predikce v časových řadách** – autoregresní modely, ARIMA[5], NS.
- **Detekce odchylek**.

3.6.2 Metodologie data miningu

Snahou je umožnit uživateli co nejsnadnější řešení. Z tohoto důvodu postupem času vznikaly metodologie z úspěšných projektů, které se praktikovaly do širšího podvědomí. Jedná se např. o SEMMA (SAS), 5A (SPSS) nebo CRISP-DM. [11]

SEMMA

Používá se k vyřešení široké škále problémů např. identifikace podvodu, obrat obchodu, předpovídání bankrotu, segmentace trhu apod. Každé písmeno názvu SEMMA reprezentuje nástroj. Jedná se o kroky, která vyvinula společnost SAS Institute. Jednotlivá písmena budou popsána [7]:

- **Sample** – konsolidace více zdrojů; úprava do vhodného formátu; ověření kvality; čistění dat.
- **Explore** – vizuální explorace; redukce dat.
- **Modify** – seskupování objektů a hodnot atributů; datové transformace.
- **Model** – automatizovaná korelace; analýza dat.
- **Assess** – porovnání modelů a interpretace; tunning dat.

[5]Anglický název: AutoRegressive Integrated Moving Average, Český název: Autoregresní integrovaný klouzavý průměr

SPSS

Jedná se o nástroj pro profesionální statistickou analýzu. Také obsahuje velké knihovny ohledně strojového učení či integraci s **big data**[6]. Zároveň obsahuje pluginy, které lze přidat k SPSS – jedná se např. o *SPSS Statistics* nebo *SPSS Modeler*. Nástroj vyvinula firma IBM a neustálého zdokonaluje. Nyní budou popsány jednotlivé body, které splňuje nástroj SPSS neboli **5A** [10]:

- **Assess** – posouzení potřeb projektu.
- **Access** – shromáždění potřebných dat.
- **Analyze** – provedení analýz.
- **Act** – přeměna znalostí na akční znalosti.
- **Automate** – převedení výsledků analýzy do praxe.

3.7 Data Lake

V českém překladu výraz *Datové jezero* – výrazy budeme považovat za ekvivalentní. Oproti DW nevyžaduje striktně strukturovaná data. Data mohou být jak nestrukturována, částečně strukturovaná nebo plně strukturována. Zároveň při použití Data Lake se zkracuje doba vývoje než jsou vidět reálné výsledky, např. nemusí se čekat půl roku, než lze přestoupit k analýze. [6]

Cílem datového jezera je co nejrychleji vytvořit reálné výsledky [6]. Navíc není možné zastavit projekt během jeho používání či ho neustále udržovat v aktuálním stavu.

K výhodám datového jezera patří rychlejší implementace. Datové jezero se využívá typicky při zkoumání dat, analýze nebo využití strojového učení. Při potřebě zjistit přesné hodnoty k určitým datům, je často využívána také varianta DW (u přibližných výsledků to neplatí) – u datového jezera se může stát, že data nebudou dostatečně pročištěna nebo budou obsahovat nekonkrétní hodnoty, které povedou ke zkreslení celkových výsledků. Většina organizací využívá obě možnosti, které jsou navzájem propojeny včetně ZČU. [29]

[6] Označení pro velká data; obsahuje data v rámci petabytů tj. 10^15 bytů

3.8 Autonomní datový sklad

Respektive Oracle Autonomous Data Warehouse (ADW) představuje službu, která funguje přes cloud DW. Slouží k automatizaci běžných úkonů, zabezpečení a tedy i k eliminaci chyb. Hlavní body, o které se autonomní datový sklad může starat [20]:

- **Autonomní správa** – umožnění provozovat výkonný, dostupný a zabezpečený datový sklad pří nízkých nákladech
  - automatické opravy / zálohování / ladění / škálování.
- **Výkon** – sledování výkonu systému a následně provedení úprav k zajištění vysokého výkonu
  - automatické indexování / správa plánů SQL / shromažďování statistik, komprese sloupců / řádků.
- **Bezpečnost** – sjednocení centra řízení zabezpečení, maskování dat, kontrola kritických databázových aktivit, objevování útoků
  - šifrování dat / klíčů, monitorování a blokování činností databáze, maskování / redigování dat.
- **Oracle Machine Learning** – algoritmy ohledně strojového učení, přenesení algoritmů do dat minimalizuje pohyb dat.
- **Grafická analytika** – slouží k odhalení skrytých vztahů v datech.
- **Prostorová analytika** – řeší formy aplikace (prostorové prac. vytížení).

4 Struktura registru CEP

IS VaVaI představuje informační systém výzkumu, vývoje a inovací, který integruje a shromažďuje data o výzkumu, vývoji a inovacích. Je to jediný oficiální autorizovaný zdroj, který je podporován z veřejných rozpočtů České republiky. Zároveň se jedná i o závazný zdroj a dle informací z registru je institucím vyplácena finanční podpora. IS VaVaI poskytuje několik nástrojů – tvorba a sběr dat, VaVER (editační rozhraní pro příjemce), ROP (administrační rozhraní pro poskytovatele podpory), veřejně přístupnou službu a v neposlední řadě aplikační rozhraní. Primární účel je hodnocení výzkumných organizací dle Metodiky 2017+. [35]

Od roku 2016 zajišťuje provoz IS VaVai **Úřad vlády České republiky** (ÚV ČR) – systém není už závislý na externích dodavatelích. [35]

Z podporovatelných zdrojů je cílem:

(a) informovat veřejnost a uchazeče o vyhlášených veřejných soutěžích VaVaI,

(b) informovat veřejnost o projektech a aktivitách VaVaI podporovaných z veřejných prostředků,

(c) informovat veřejnost o aktivitách ve výzkumu, vývoji a inovacích,

(d) informovat i další orgány a osoby stanovené zvláštními právními předpisy,

(e) kontrolovat poskytování a používání účelové podpory VaVaI,

(f) příprava návrhu státního rozpočtu. [35]

IS je provozován na volně dostupných technologií – linuxový server s webových serverem Apache, server side jazyk PHP7 a databáze MYSQL. [35]

Celkem obsahuje čtyři vzájemně propojené oblasti: CEP (Centrální evidence projektů), CEA (Centrální evidence aktivit VaVaI), RIV (Rejstřík informací o výsledcích) a VES (Evidence veřejných soutěží ve VaVaI). [35]

CEP obsahuje údaje o projektech a poskytovanou účelovou podporu. Jednotlivé údaje jsou uvedeny ve třech fázích: při zahájení, během řešení a po ukončení řešení. Pro upřesnění, **CEP** obsahuje údaje určující projekt – název, předmět řešení, příjemci, řešitelé, údaje o uzavření smlouvy, kategorie výzkumu, poskytovatel podpory, program a mnohem více. [35]

4.1 Application Programming Interface (API)

Od roku 2019 funguje API v ostrém provozu. Mezi základní atributy patří:

- formát výstupu JSON a JSONP,
- kódování UTF-8,
- odesílání dotazu do API pouze přes POST,
- pracovní doba 24/7,
- zabezpečení tokenem a kontrolou IP adresy příchozího požadavku,
- možnost volby rozsahu a stránkování obsahu,
- lze použít více tokenů s různým oprávněním,
- rozsah dat dle konfigurace obsahuje všechny informační oblasti – CEA, VES, CEP, RIV. [35]

Při získání tokenu je nutné se ozvat přímo na emailovou adresu[1] IS VaVaI. Token má korelaci s IP adresou – omezení na jednu IP adresu. V případě provolávání je nutné vyplnit povinné parametry (obrázek 4.1) – token, oblast a režim. V případě potřeby upřesnění výsledné hodnoty je nutné navolit i zbývající parametry. V případě, že provolání skončí s chybou, která indikuje, že dat je přespříliš, tak lze provolání omezit na počet stránek / limitací / upřesnění pomocí číselníku. [38]

Zde je zobrazena základní struktura odpovědi (obrázek 4.2), která se skládá z hlavičky a z dat. Pří úspěšném provolání příkazu se objeví navíc data, která se nachází pod hlavičkou. Provolané příkazy si lze zobrazit na stránce `https://api.isvavai.cz/` v přehledu logu API pro TOKEN. Po zadání tokenu se objeví přehledná tabulka, která obsahuje datum zadání příkazu, odkud byl příkaz zadán, kód chyby, oblast, režim, kolik parametrů bylo zadáno, počet výsledků a jak dlouho se příkaz zpracovával.

Následuje ukázka provolání testovacího příkazu ke zjištění funkčnosti. Jsou vyplněné parametry – token:retezec, oblast:cep, rezim:test. Provolání příkazu lze reprodukovat několika způsoby. Nejjednodušší provolání lze provést přímo pomocí PHP požadavku – samotná ukázka požadavku je uvedena v dokumentu: **Specifikace API pro IS VaVaI** [38] – dokument lze stáhnout ze stránek `https://api.isvavai.cz/` ve hlavičce **Dokumentace**

[1] isvavai@vlada.cz

API ke stažení. Pro lepší přehlednost byl zvolen nástroj **Insomnia**, který umí velice jednoduše a intuitivně provolávat požadavky z API. Jako výsledek úspěšného provolání je výstup 4.1.

Popis / význam parametru	Parametr	Povolená hodnota
Autorizační token	token [P]	řetězec, 40 znaků
Formát odpovědi (výchozí: json)	format	"json", "jsonp"
Callback fce formátu JSONP (výchozí: func)	callback	řetězec bez mezer a diakritiky
Identifikátor informační oblasti	oblast [P]	"csl", "cea-pos", "cea-prg", "cea-subj", "cea-vvi", "ves", "cep", "riv", "cez"
Režim práce / výstupu	rezim [P]	"test", "ciselnik", "kontrolni-cislo", "filtr-seznam", "filtr-detail"
Počet záznamů na stránku (výchozí: 1 / 0 *)	limit	kladné celé číslo 0 - vypnuté stránkování
Strana dle počtu výsledků a limitu (výchozí: 1)	strana	kladné celé číslo
Formát financí - zaokrouhlení (výchozí: 1) ** [F] - označení hodnot ovlivněných volbou v dokumentaci	finance	0 - částky v Kč s přesností na 2 des. m. 1 - částky zaokrouhlené na tisíce Kč

* Výchozí hodnota stránkování přes den nastavena na 1 a nelze ji změnit, pro večení režim je uživatelsky konfigurovatelná.

** Pro zachování kompatibility s IS, které pracují s původním formátem financí zaokrouhlených na celé tisíce Kč je výchozím stavem a výstupem všech financí v IO CEA, CEP, CEZ a VES zaokrouhlená částka. Finance v tisících Kč jsou automaticky zaokrouhleny z částky v jednotkách Kč s přesností na 2 desetinná místa. Pokud chcete získávat informace v přesných finančních částkách, nastavte parametr finance=0.

Obrázek 4.1: Základní parametry dotazu [38]

Atributy hlavičky	Popis atributů hlavičky
kod	Kód chybové zprávy generované API
zprava	Zpráva API odpovídající chybovému kódu
pocet-vysledku	Celkový počet výsledků vybraných na základě zadaných parametrů
limit	Aktivní limit pro počet výsledků na jedné straně
pocet-stran	Celkový počet stran výsledků
strana	Číslo stránky, ze které jsou data poskytnuta
verze	Aktuální verze API

Obrázek 4.2: Základní struktura odpovědi [38]

```
{
   "hlavicka": {
      "kod": 100,
      "zprava": "Test funkce serveru OK",
      "pocet-vysledku": 0,
      "limit": 1,
      "pocet-stran": 1,
      "strana": 1,
      "finance": 1,
      "verze": "2.0.1"
   },
   "data": []
}
```

Kód 4.1: Test provolání serveru API (zdroj vlastní)

Pro práci s daty bylo po přezkoumání API nutné stáhnout všechny detaily a příslušné číselníky, které se naparsují na hlavní data (detail).

4.2 Data detail

Na provolání příkazu je nutné si požadavek zmenšit z důvodu obsáhlosti dat. Požadavek si upřesníme pomocí fitru `fp-soutez-druh`, který může dosahovat hodnot M2, OP, RP, VL, VS a VZ. V případě obsáhlosti dat je vhodné upřesnit parametry **strana** a **limit**. Ukázka parametru může vypadat např. jako kód 4.2:

```
(
   'token'          => 'mujToken',
   'oblast'         => 'cep',
   'rezim'          => 'filtr-detail',
   'fp-soutez-druh' => 'M2',
   'limit'          => 100,
   'strana'         => 2
);
```

Kód 4.2: Ukázka parametru provolání - detail (zdroj vlastní)

Po stáhnutí všech detailů byla snaha všechny data spojit do jednoho velkého molochu dat pro snazší práci. Ukázka úspěšného provolání 4.3 – obsahuje jenom část informace výstupu z důvodu přehlednosti.

```
{
   "hlavicka": {
   "kod": 200,
   "zprava": "Uspesne dokonceno",
   "pocet-vysledku": 100,
   ...},
   "data": [
      {  "kod": "7AMB15AR001",
         "kod-duvernosti": "S",
         "nazev": "Vlivy zmeny klimatu na horke vlny a pravdepodobnosti
    jejich opakovani",
         "nazev-anglicky": "Climate change effects on heat waves and
    their recurrence probabilities",
         ...,
         "finance": {
            "tabulka": {
               "stav": {
                  "2015": "CER",
                  "2016": "CER",
                  "2017": ""
               }, "CEL": {
                  "2015": 140,
                  "2016": 23,
                  "2017": 0
               }, "SRU": {
                  ...
               }, "VZZ": {
                  ...
               }, ...
            }, "rok-od": 2015,
            "rok-do": 2017
         },
         ...
      }
   ],
   ...
}
```

Kód 4.3: Ukázka odpověď - detail M2 (zdroj vlastní)

4.3 Data číselníky

Po zhlédnutí specifikace v dokumentaci bylo možno zjistit, jaké číselníky jsou potřeba, a které naopak potřebné nejsou. Pro snazší orientaci je ve specifikaci API určeno **ID** číselníku. Bylo nutné vybrat potřebná data, se

kterými chceme pracovat. Může se zvolit varianta stahování všech číselníků nebo se stáhnout jen ty číselníky, které víme jistě, že se budou používat. My jsme volili variantu stažení vybraných číselníků.

Nyní bude následovat výčet číselníku, které půjdou propojit s hlavní tabulku detailu. [38]

- **Druh soutěže** – popisuje, o jaký typ soutěže se jedná; např. M2: dvoustranná dohoda o mezinárodní spolupráci.
- **Hodnocení** – zhodnocuje, jestli daný projekt dopadl úspěšně, neúspěšně či s připomínkami.
- **Kategorie výzkumu a vývoje** – o jakou kategorii projektu se jedná; např. AP: Aplikovaný výzkum, IN: Inovace.
- **Klasifikace oboru** – do jaké klasifikace se dá projekt zařadit; např. AB: Dějiny, BJ: Termodynamika, CE: Biochemie, IN: Informatika.
- **Kód důvěrnosti** – jak moc je daný projekt důvěrný (tajný).
- **Kódy zemí** – číselník na země; např. AW: Aruba, CZ: Česká republika.
- **Poskytovatel** – tento číselník se nevyskytoval v API a musel se dohledat na stránkách IS VaVaI, kde byl dostupný ve formátu .csv.
- **Role příjemce** – o jakého příjemce vůči projekt se jedná; např. K: subjekt je garantujícím příjemcem, SV: subjekt je dalším účastníkem.
- **Role řešitele** – jestli se jedná o prvotního nebo další řešitele.
- **Skupiny oboru** – zařazení do širší skupiny oboru; např. K: vojenství.
- **Stav průběhu řešení** – o jaký projekt z časového hlediska se jedná.
- **Subjekt** – druh subjektu, který pracuje na projektu; např. VOP CZ.
- **Veřejné zakázky** – stejné jako číselník Druh soutěže ale popsán stručněji a ochuzen o jeden druh soutěže.

Většina číselníků má stejnou strukturu až na pár číselníků, které mají specifickou strukturu – jedná se o číselníky: **Velké výzkumné infrastruktury** (9997), **Vědní obory OECD** (9998) a **Patentové úřady** (9999). Ukázka volání běžného číselníku **druh soutěže** 4.4:

```
(
   'token'          => 'mujToken',
   'oblast'         => 'csl',
   'rezim'          => 'ciselnik',
   'id'             => 1202
);
```

Kód 4.4: Ukázka parametru provolání - číselník druh soutěže (zdroj vlastní)

Odpověď číselníku **druh soutěže** 4.5:

```
{
   "hlavicka": {
      "kod": 200,
      "zprava": "Uspesne dokonceno",
      "pocet-vysledku": 7,
      ...
   },
   "data": [
      {
         "stav": 1,
         "kod": "M2",
         "popis-cz": "Dvoustranna dohoda o~mez. spolupraci",
         "popis-en": "",
         "xml": "",
         "kod-alt": ""
      },
      ...
   ]
}
```

Kód 4.5: Ukázka odpověď - číselník druh soutěže (zdroj vlastní)

4.4 Nástroj k dotazování v API

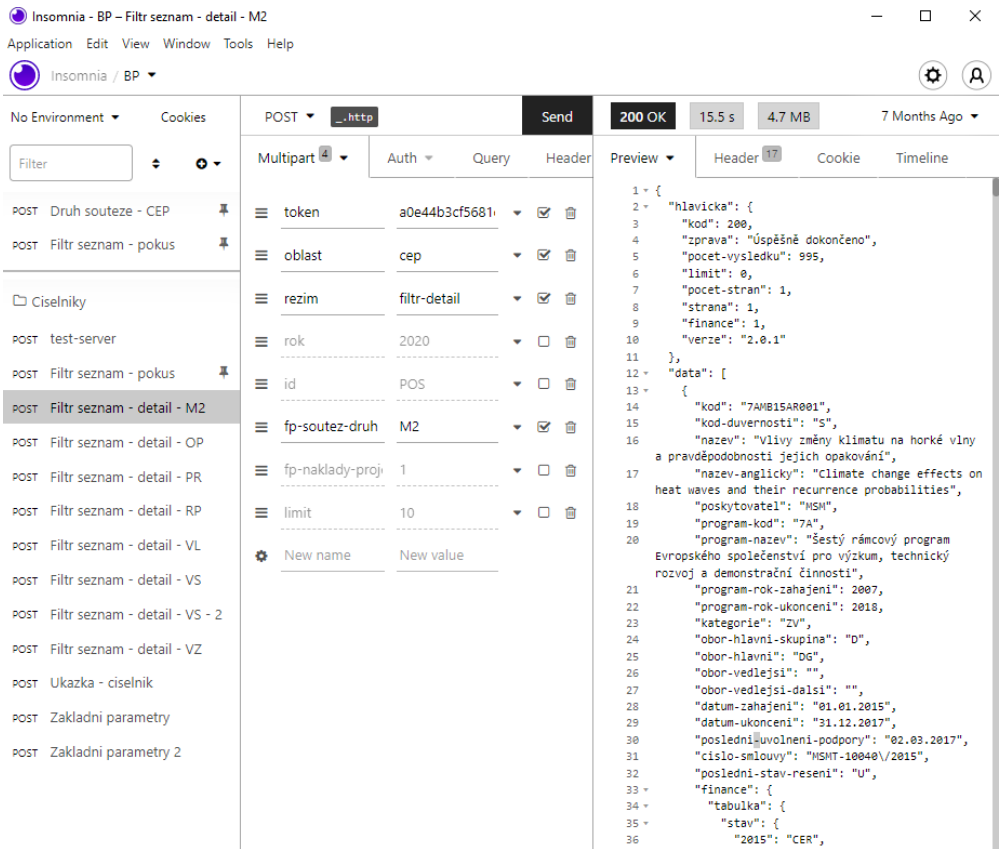
Po konzultacích a zjišťování vhodnosti nástrojů podporujících REST/SOAP služby byl zvolen nástroj Insomnia. **Insomnia**[2] je nástroj, který slouží pro dotazování a navrhování API naložených na HTTP požadavcích. Výhodou je snadná obsluha obsahující pokročilé funkce jako generování kódu, nastavování proměnných či automatická autentizace. Lze si také nastavit více prostředí, a tím rozdělit testovací od vývojového prostředí, import/export dat nebo vývoj vlastních pluginů.

Pro práci s API byla použita desktopová verze[3], která je vidět na obrázku 4.3.

[2] `https://insomnia.rest/`

[3] Možnost stažení na platformy: Windows, Linux nebo MacOS

Byl zvažován známější nástroj **Postman**, který obsahuje více funkcionalit, ale z důvodu větší přehlednosti byl upřednostněno prostředí Insomnia. Jestliže by se jednalo o minimální počet požadavků na API, tak existuje možnost provolávání přes PHP skript (ukázka je uvedena v dokumentu *Specifikace API pro IS VaVaI* [38]), ale pro větší přehlednost byl upřednostněn nástroj Insomnia.



Obrázek 4.3: Ukázka provolání - Insomnia (zdroj vlastní)

4.4.1 Stažení dat

Pro přístup k datům je, jak bylo zmíněno, nutné vlastnit autorizační token. Následně je vhodné vymezení potřebných dat. Pro prvotní zhotovení datové kostky nejsou vybrána jen data, která budou datovou kostku plnit. Z toho důvodu byl aplikován způsob stažení více dat a separace proběhla následně až během plnění datové kostky – data byla omezena pouze na množinu CEP.

Nyní byl aplikován způsob procházení dokumentu Specifikace API[4] v podkapitole **Číselníky** a **Informační oblast CEP**. Data, která lze stáhnout jsou popsané v každé podkapitole v odstavci **Definice dotazu** (ukázka na Obrázku 4.4). Odpověď dotazu je rovněž popsána v odstavci **Struktura odpovědi dotazu**. V případě, že by bylo třeba data filtrovat hned voláním z API, obsahuje dokumentace kapitoly **Definice filtračních parametrů**, který toto umožňuje. Tato varianta nebyla aplikována z důvodu neznalosti dat a upřednostnění filtrace až po transformaci dat v databázi.

2.8.1 Definice dotazu na kontrolní čísla - CEP

Kategorie se zvolí automaticky, dle rozsahu zadaných parametrů dotazu. Režim získávání kontrolních čísel není limitován počtem výsledků na jeden dotaz, pouze min. časem od posledního dotazu 1s.

Kategorie [prefix] **/ Název parametru**	**Parametr**	**Povolená hodnota**
Konkrétní kontrolní číslo		
Identifikátor projektu	projekt-id	řetězec
Rok sběru dat	rok	číslo, 4 cifry
Seznam kontrolních čísel z jednotlivých let řešení projektu		
Identifikátor projektu	projekt-id	řetězec

2.8.2 Struktura odpovědi dotazu na kontrolní čísla - CEP

Kategorie (nás.) **/ Atribut**	**Název atributu**
Konkrétní kontrolní číslo (1)	
kontrolni-cislo	Kontrolní číslo pro konkrétní projekt a rok
Seznam kontrolních čísel z jednotlivých let řešení projektu (n)	
rok	Rok sběru fáze projektu
kontrolni-cislo	Kontrolní číslo pro příslušný rok

2.8.3 Definice filtračních parametrů - CEP

Kategorie [prefix] **/ Název parametru**	**Parametr**	**Povolená hodnota**
Projekt [pr]		
Identifikační kód	pr-kod	řetězec
Název projektu	pr-nazev	řetězec
Financování projektu [fp]		
Druh souteže	fp-soutez-druh	číselník, id: 1202
Identifikační kód	fp-soutez-kod	řetězec
Poskytovatel	fp-poskytovatel	číselník, id: POS
Celkové náklady v tis. Kč / Kč - od	fp-naklady-cel-od	číslo / desetinné číslo [F]
Celkové náklady v tis. Kč / Kč - do	fp-naklady-cel-do	číslo / desetinné číslo [F]

Obrázek 4.4: Definice dotazu, odpovědi a filtračních parametrů [38]

[4]Celý název dokumentu: Specifikace API pro IS VaVaI

5 Transformace dat v databázi

V návaznosti na předešlou kapitolu bude nejdříve popsáno provedení nahrání dat formátu JSON (JavaScript Object Notation) do Oracle databáze. Následně bude vykonáno čistění dat za pomocí Oracle funkcí. Budou následovat informace o čistění dat a tvorbě package, který se stará o vytvoření a naplnění datové kostky.

5.1 Využité technologie

Pro nahrávání / importování, modelování až po exportování dat se využívaly primárně dva nástroje: **Oracle SQL Developer** a **Sublime text**. Byly zváženy i podobné nástroje, které by zvládly práci lépe, ale kvůli komerčnosti se zůstalo u těchto nástrojů. Jedná se např. o nástroj **Toad**, který by kompletně nahradil Oracle SQL Developer – navíc přichází s mnoha funkcemi navíc a podle uživatelských recenzí výrazně vede oproti konkurenci. Jde však o placenou variantu. Podobný nástroj, který vypadá uživatelsky přívětivě je DataGrip od firmy JetBrains. Po porovnání s konkurencí přesto nejlépe vycházel nástroj Toad, ale kvůli komerčnosti tento nástroj nebyl využit.

Oracle SQL Developer

Jde o freeware (bezplatný nástroj) firmy Oracle[1]. Jedná se o základní, ale robustní sadu nástrojů. Je vhodný pro všechny činnosti nad DB od klasické správy databáze, tvoření PL/SQL skriptů, modelování až po migraci z/do třetích stran v rámci Oracle i mimo něj.

Sublime text

Jedná se o editor textových souborů, které byl hojně užíván během práce s Oracle databází. Během tvoření nových tabulek, editací nebo hledání klíčových slov pomocí regexů[2] byl nedílnou součástí, a to z důvodu přívětivějšího prostředí, funkce zkratek nebo funkcionalit, které nástroj Oracle developer nepodporuje.

[1] Ke stažení: `https://www.oracle.com/tools/downloads/sqldev-downloads.html`
[2] Čerpané inspirace z tvoření regexu: `https://regexone.com/`

5.2 Překlopení dat do databáze

Při nahrávání z formátu JSON do databázové struktury bylo nutné prozkoumat strukturu číselníku a detailu. Tento proces lze rozdělit na několik menších úkolů – tj. **vytvoření tabulek** o stejné struktuře jako formát souboru ve formátu JSON, **transformace** během nahrání dat do nově vytvořené tabulky, **čistění dat** a **vytvoření závislosti** mezi ostatními tabulkami. V první řadě bylo nutné si stanovit stejnou štábní strukturu týkající se pojmenování souborů. Ve druhém kroku byl zjištěn nejlepší případ nahrávání dat formátu JSON do databáze. V posledním kroku, bylo spolu s čistěním dat zajištěno smysluplné provázání číselníku s hlavními daty.

5.2.1 Tvorba tabulky

Během procesu získávání dat se ukázalo snadnější nahrávat menší soubory – v tomto případě zpravidla číselníky. Při pohledu na referenční číselník `Druh_souteze` struktura vypadá jednoduše – bylo potřeba vytvořit pouze sloupce `kod`, `stav`, `popis_cz`, `popis_en`, `xml` a `kod_alt`.

Pro pořádek byly všechny číselníkové tabulky pojmenovány s předponou **CIS**, tedy celý název tabulky je `CIS_DRUH_SOUTEZE`. Pojmenování atributu č. 1 má odkazovat na primární klíč (Primary Key, PK) – v tomto případě se jedná o atribut `kod`, který je unikání a proto obdržel pro snazší identifikaci předponu **ID**. Zbytek slova byl tvořen názvem tabulky. Celý název PK je `ID_DRUH_SOUTEZE`. Ostatní atributy se vyskytují bezprostředně za atributem č. 1. Ukázka vytvořené tabulky reprezentující číselník 5.1:

```
CREATE TABLE cis_druh_souteze (
   id_druh_souteze         VARCHAR2(8 CHAR) PRIMARY KEY,
   stav                    NUMBER(16),
   popis_cz                VARCHAR2(512 CHAR),
   popis_en                VARCHAR2(512 CHAR),
   xml                     VARCHAR2(64 CHAR),
   kod_alt                 VARCHAR2(64 CHAR)
);
COMMENT ON COLUMN cis_druh_souteze.id_druh_souteze
   IS 'zkratka souteze na 2 pismena';
COMMENT ON COLUMN cis_druh_souteze.STAV
   IS '1=platne; 0=neplatne';
COMMENT ON COLUMN cis_druh_souteze.POPIS_CZ
   IS 'cely nazev v~ceskem jazyce'; -- ...
COMMENT ON COLUMN cis_druh_souteze.POPIS_EN
   IS 'cely nazev v~anglickem jazyce';
```

Kód 5.1: Vytvoření tabulky: `CIS_DRUH_SOUTEZE` (zdroj vlastní)

5.2.2 Nahrání dat do databáze

Byly vyzkoušeny celkem tři přístupy nahrávání dat do RDB a jeden přístup byl zvážen. První přístup se snažil o automatické nahrání přes nástroj **Oracle Apex**. Druhý přístup byl aplikován **programově**, kdy se jednalo o napsání skriptu spolu se souborem obsahující formát JSON. Třetí přístup od druhého se moc nelišil. Jediný rozdíl byl, že data byla nahrána na server a odsud se volaly skripty z vývojového prostředí. Poslední zvažovanou možnosti byla konverze dat do RDB pomocí automatického nástroje [37].

Oracle Apex

Snaha nástroje je zmenšit programovací oblast uživatele na co nejmenší část a zvýšit samostatnou produktivitu uživatele[3]. Převážná většina událostí vzniká za pomocí „naklikávání“. [1]

Nástroj byl vyzkoušen kvůli možnosti nahrání souborů formátu JSON do RDB. Při vyzkoušení souborů o velké velikosti, které obsahovaly přes 100 a více atributů program nebyl schopen zpracovávat data – zamítnuté zpracování. Při práci pod 100 atributů dokázal nástroj úspěšně přeložit data do RDB, ale formát dat, který byl vyžadován (případně osekán), nebyl k dispozici. Bylo by nutné věnovat ještě velké úsilí, aby se vytvořil korektní formát. Nakonec se od tohoto přístupu upustilo.

Programově bez adresáře

Tento přístup sloužil primárně k seznámení se s formátem JSON a jeho převodu do RDB. V první řadě byla vytvořena speciální tabulka, která obsahovala dva údaje – PK a data (obsah formátu JSON). Následně pomocí SQL příkazu se vložila data z formátu JSON do nové tabulky. V dalším kroku se pomocí příkazů pro výběr dat z tabulky dala data upravit a vložit do nové cílové tabulky, která byla připravena pro data v podobě, kterou potřebujeme. Pro ukázku je uvedeno volání příkazu `SELECT` 5.2 nad tabulkou `departments_json`, která obsahuje data ve formátu JSON. [22, 27]

[3]Podrobnější info: `https://apex.oracle.com/`

```
SELECT s.*
FROM  departments_json dep, JSON_TABLE (
         dep.department_data, '$' COLUMNS (
            department PATH '$.department',
            NESTED PATH '$.employees[*]'
            COLUMNS (
               NAME PATH '$.name',
               JOB  PATH '$.job'
   ) ) ) s
WHERE  dep.department_id = 110;
```

Kód 5.2: `SELECT` nad tabulkou `departments_json` (zdroj vlastní)

Problém se naskytl, když byla potřeba nahrát větší soubor, než je velikost 32KB pomocí PL/SQL[4] (pomocí SQL je omezení na 4000 bytů). Tohle omezení je zcela srozumitelné, protože SQL Developer nepočítá s takovým rozsáhlým skriptem. Řešením bylo použití tzv. „chunků“ 5.3, které se vnitřně překonvertují do datového typu CLOB (character large object). Soubory z CEP mají ovšem řádově velikosti MB/GB/TB. Z tohoto důvodu byl pro uložení využito adresáře (directory) na serveru, na kterém DB běží.

```
INSERT INTO json_data (id, raw_json, description)
VALUES (sys_guid(),
   to_clob('{"hlavi') || to_clob('cka": { "kod": 200,'),
   'cast formatu JSON');
```

Kód 5.3: Ukázka použití chunků při vkládání JSON do tabulky (zdroj vlastní)

Programově za pomocí adresáře

Nejdříve bylo potřeba nahrát veškeré soubory na školní server do adresáře, ke kterému je přístup umožněn – adresář: `/home/oracle/JEZDO`. Následně byl vytvořen **DIRECTORY** pomocí příkazu `CREATE OR REPLACE DIRECTORY CEP_FILTR_DETAIL AS '/home/oracle/JEZDO';` (příkazy vyžadují oprávnění – nutno požádání administrátora).

V dalším kroku byly využity znalosti nahrávání formátu JSON do relační databáze [22]. Proces vkládání dat do předem vytvořené tabulky se skládá z volání funkce `JSON_TABLE`, která obsahuje dva parametry. První je místo, odkud jsou data vybrána – v našem případě volaná funkce `BFILENAME`, s parametry název directory **CEP_FILTR_DETAIL** a názvem souboru tj.

[4] Procedural Language/Structured Query Language

Druh_souteze-CEP....json [23]. Druhý parametr tvoří už samostatný výběr ze souboru, kde označení `$.data[*]` značí zanoření do formátu JSON. Označení `COLUMNS` se stará o výběr konkrétních atributů a jejich uložení do tabulky (viz. zdrojový kód 5.4).

```
INSERT ALL INTO cis_druh_souteze
SELECT *
FROM JSON_TABLE (
   BFILENAME('CEP_FILTR_DETAIL', 'Druh_souteze-CEP-1635977209403.json'),
   '$.data[*]'
   COLUMNS(
      id_druh_souteze      VARCHAR2(8 CHAR)    PATH '$."kod"',
      stav                 NUMBER(16)          PATH '$."stav"',
      popis_cz             VARCHAR2(512 CHAR)  PATH '$."popis-cz"',
      popis_en             VARCHAR2(512 CHAR)  PATH '$."popis-en"',
      xml                  VARCHAR2(64 CHAR)   PATH '$."xml"',
      kod_alt              VARCHAR2(64 CHAR)   PATH '$."kod-alt"'
   )
);
```

Kód 5.4: Nahrání dat z číselníku do tabulky `CIS_DRUH_SOUTEZE` (zdroj vlastní)

Výzvou začíná být kombinace konstrukce formátu JSON s jazykem PL/SQL a její následná transformace do tabulky. Pro ukázku je uveden zdrojový kód 5.5. Jedná se o skript, který má na starost výběr dat přes více souborů a zároveň má větší hloubku zanoření dat. Druhá část skriptu se stará o stažení dat z atributů – jedná se o atribut „`$."dalsi-prijemci"[*]`“ místo atributu „`$.prijemci.resitele[*]`“.

```
DECLARE
   v_number  NUMBER := 0;
   v_path_of_file VARCHAR(60);
   TYPE array_t IS
   VARRAY(4) OF VARCHAR2(25);
   v_array   array_t := array_t('M2-1635976645346', 'OP-1635976750703', '
    PR-1635976763801', 'RP-1635976772444'); -- pole bylo zmenseno
BEGIN
   v_number := v_array.count - v_number;
   FOR i IN 1..v_number LOOP
      v_path_of_file := 'filtr-seznam_detail_' || v_array(i) || '.json';
      dbms_output.put_line('Processing␣file:␣' || v_path_of_file);

      ---- INSERT_06 ----
      INSERT  ALL INTO prijemci_tmp
      SELECT  *
      FROM JSON_TABLE (
         BFILENAME('CEP_FILTR_DETAIL', v_path_of_file),
```

```
18          '$.data[*]'
19          COLUMNS(
20            id_ai          NUMBER              PATH '$."someNoSense"',
21            kod            VARCHAR2(64  CHAR) PATH '$."kod"',
22            prijemce_id    VARCHAR2(64  CHAR) PATH '$."prijemci"."id"',
23            prijemce_role  VARCHAR2(8   CHAR) PATH '$."prijemci"."role"',
24            prijemce_nazev VARCHAR2(128 CHAR) PATH '$."prijemci"."nazev"',
25            NESTED PATH '$.prijemci.resitele[*]'
26            COLUMNS (
27             role        VARCHAR2(8   CHAR) PATH '$."role"',
28             titul_pred  VARCHAR2(64  CHAR) PATH '$."titul-pred"',
29             ...
30             )
31          )
32       );
33    END LOOP;
34 END;
35 );
```

Kód 5.5: Nahrání dat z filtru detailu do tabulky `PRIJEMCI` (zdroj vlastní)

Automatická Konverze z NoSQL do Relační Databáze

Podle výsledků z odborné publikace [37] se nástroj choval velice obdivuhodně. Tato možnost vyzkoušení byla zvážena – musela by se stažená data nahrát opětovně do NoSQLa nebo požádat o přístup IS VaVaI do NoSQL. Po zvážení, jaké všechny problémy by se mohly naskytnout během realizace, zůstalo u ručního programového zpracování za pomoci adresáře.

5.2.3 Určení závislostí

Po nahrání všech souborů do DB bylo vhodné určit závislosti mezi hlavní tabulkou `GEN_DATA`, číselníky, příjemci a financemi. Tento krok usnadnil práci při tvorbě datové kostky, kdy nebylo třeba opětovně prohledávat veškeré tabulky a zjišťovat závislosti mezi nimi.

Pro urychlení hledání závislostí byl použit dokument specifikace [38] a tabulky, které nejsou číselníky – zde bylo nutné pro každý atribut projít veškeré hodnoty a spojit je s příslušnými číselníky popřípadě i příjemci, financemi. Příslušné závislosti lze nalézt v tabulce `GEN_DATA`, ale i mezi ostatními tabulkami.

5.3 ETL

Velká část procesu byla provedena v rámci stažení a nahrání dat. V tomto kroku bude rozebrané drobné čistění, které bylo nutné provést a vytvoření pomocných tabulek, které byly nutné pro naplnění datové kostky.

Až na některé výjimky se většina tabulek do DB nahrála bez chyb. Tabulky, které obsahovaly chyby v datumu, se pomocí konverzní funkce `TO_DATE()` převedly na správný formát.

Tabulka `GEN_DATA` obsahovala chybu v atributu `DATUM_ZAHAJENI` nebo atributu `DATUM_UKONCENI`, kde se vyskytoval pouze rok namísto den + měsíc + rok. Toto bylo Vyřešeno přidáním defaultního datumu před rok `01.01.` při zahájení nebo `31.12.` při ukončení.

Další z nedostatků se objevil z „nedokonalých“ dat, kde data nebyla ucelená – obsahovala tabulátory a entery. Ty samy osobě problém během práce s daty nedělaly, ale při exportování dat do .csv formátu se soubory tvářily „nafoukle“ – vyřešeno pomocí funkce `TRANSLATE()`, kde se přebytečné znaky nahradily prázdným charakterem.

Při nahrávání všech dat (kromě číselníku) se musela provést kontrola duplikátů. V řádu jednotek se stalo, že v datech byly duplikáty. Duplikáty byly odstraněny kvůli vytvoření PK z atributu `KÓD`. Tabulka `PRIJEMCI` obsahovala duplicitní kód projektu, příjemce i účastníka (VEDIDK) – zde byl vytvořen PK pomocí volání `id_ai = ROWNUM`, kde se na atribut id_ai, který neobsahoval žádná data, přepsala číselná hodnota, která se vytvořila podle toho, na jakém řádku se záznam nacházel.

Pro zkušební účely byly vytvořené procedury, které se staraly o vytvoření tabulek, jež reprezentovaly dimenze a faktovou tabulku. Tento proces jenom demonstroval tvoření celé funkční struktury. V kapitole **Tvorba datové kostky** bude popsán automatizovaný postup nahrávání a aktualizovaní datové kostky pomocí package.

5.4 Pomocné tabulky

Byly vytvořené pomocné tabulky `DIM_ROK` a `FIN_CEP_POMOCNA`, které slouží k vytvoření datové kostky.

5.4.1 `DIM_ROK`

Tabulka byla vytvořena za účelem filtrování roků v tabulce faktů – obsahuje pouze ID PK a posledních 12 roků od dnešního roku (použití SYSDATE).

5.4.2 FIN_CEP_POMOCNA

Slouží k seskupení čtyř tabulek (FIN_STAV, FIN_CEL, FIN_SRU, FIN_VZZ) týkající se financí. Tímto krokem se ulehčila práce během tvoření datové kostky. Tabulka obsahuje celkem šest atributů, kde zbylé dva označují kód a rok – tabulka sice obsahuje mnohem více záznamů, ale pro použití v datové kostce je tento krok nezbytný. Pro lepší ukázku je demonstrována tabulka 5.1.

KOD	FIN_STAV	FIN_CEL	FIN_SRU	FIN_VZZ	ROK
2C06008	CER	5005	3753	0	2010
2C06008	CER	2687	1855	0	2011
8A20001	PRI	709	284	213	2021
EA 4.2PT03	PLA	44114	17645	0	2015

Tabulka 5.1: Tabulka FIN_CEP_POMOCNA (zdroj vlastní)

Původní tabulky jsou navíc hodně řídké. Pro lepší ukázku je uvedena tabulka FIN_STAV 5.2.

ID_KOD	YEAR_2010	...	YEAR_2021	ROK_OD	ROK_DO
2C06008	CER	...	NULL	2008	2017
8J19AT030	NULL	...	PRI	2019	2021
MEB021041	CER	...	NULL	2010	2011
JBR/1/00	NULL	...	NULL	1999	2002

Tabulka 5.2: Tabulka FIN_STAV (zdroj vlastní)

5.5 Tvorba datové kostky

Pro automatizované vytvoření a následně aktualizování datové kostky byl vytvořen package s názvem PG_CEP. Package musí obsahovat jak „HEAD“, tak i BODY. Hlavička package obsahuje pouze možné volané procedury v BODY.

Jednotlivé volání se provádí pomocí EXEC PG_CEP.<NAZEV_PROCEDURY>, kde za název procedury lze zvolit libovolnou proceduru z hlavičky. Nutno dodat, že pro správnou funkčnost package se předpokládá vytvořené tabulky reprezentující dimenze a faktovou tabulku.

5.5.1 PACKAGE PG_CEP

Vytvoření hlavičky package začíná příkazem `create or replace PACKAGE PG_CEP AS` a končí `END PG_CEP;`. Jednotlivé nástrahy procedur budou stručně popsány:

- `PROCEDURE dim_druh_souteze_akt` – atribut `verejna_soutez` může nabývat hodnoty NULL. Vyřešeno pomocí funkce `NVL()`.
- `PROCEDURE dim_hodnoceni_clear` – děje se tak u vkládání dat do dimenzí, kde je tzv. natural key (přirozený klíč, kterým se faktová tabulka při naplňování připojuje k dimenzi nebo je skript velice náročný a merge by zahltil celou paměť) téměř každý atribut. Tabulku je nutno nejprve vyčistit a nahrát vše znova (většina neprimárních klíčů musí být v podmínce `WHEN MATCHED THEN`).
- `PROCEDURE dim_hodnoceni_akt` – vkládá data do prázdné tabulky.
- `PROCEDURE dim_kategorie_akt` – funkčnost bez problému.
- `PROCEDURE dim_obor_akt` – data z číselníku jsou rozšířená pro všechen `typ_oboru` pomocí `UNIONu`.
- `PROCEDURE dim_obor_skupina_akt` – funkčnost bez problému.
- `PROCEDURE dim_poskytovatel_akt` – funkčnost bez problému.
- `PROCEDURE dim_prijemce_akt` – atributy `prijemce_id` a `prijemce_role` můžou nabývat hodnoty NULL. Vyřešeno pomocí funkce `NVL()`.
- `PROCEDURE dim_program_akt` – funkčnost bez problému.
- `PROCEDURE dim_rok_akt` – kontroluje datum posledních 12 let.
- `PROCEDURE dim_stav_reseni_akt` – funkčnost bez problému.
- `PROCEDURE fact_fin_pomocna_clear` – vyprázdní tabulku.
- `PROCEDURE fact_fin_pomocna_akt` – popis naplnění tabulky je popsán v podkapitole `FIN_CEP_POMOCNA`.
- `PROCEDURE fact_cep_clear` – vyčistí faktovou tabulku.
- `PROCEDURE fact_cep_akt` – naplní faktovou tabulku.
- `PROCEDURE cep_kostka_akt` – spustí všechny výše popsané procedury.

5.5.2 PACKAGE BODY PG_CEP

Jednotlivé aktualizace dimenzí probíhají pomocí konstrukce MERGE (zdrojový kód 5.6), kde se data, která nejsou v tabulce, vloží a data, která se vyskytují, se pouze aktualizují.

```
PROCEDURE dim_kategorie_akt IS
  v_my_name CONSTANT VARCHAR2(100 CHAR) := 'AKTUALIZACE␣DIM_KATEGORIE:␣';
BEGIN
   MERGE INTO dim_kategorie t
   USING (SELECT DISTINCT  id_kategorie,
                     stav,
                     popis_cz
                     FROM cis_kat_vyzkumu_vyvoje
        )s
   ON (t.kod = s.id_kategorie)
   WHEN MATCHED THEN
      UPDATE
         SET t.stav = s.stav,
            t.popis_cz = s.popis_cz
   WHEN NOT MATCHED THEN
      INSERT (
         t.kod,
         t.stav,
         t.popis_cz
      )
      VALUES (
         s.id_kategorie,
         s.stav,
         s.popis_cz
      );

   DBMS_OUTPUT.put_line(v_my_name || SQL%ROWCOUNT);
END;
```

Kód 5.6: Aktualizace dat pomocí MERGE v tabulce DIM_KATEGORIE_AKT (zdroj vlastní)

Po aktualizování všech dimenzí přichází na řadu naplnění faktové tabulky. Před plněním tabulky se musí samotná tabulka pročistit. Faktová tabulka obsahuje atributy jako FK odkazující do dimenzí (IDK), neobsahuje PK (tento údaj při analýze by se nevyužil) a fakta (např. název projektu, datum zahájení, datum ukončení a další). Jednotlivé propojování faktové a dimenzionální tabulky se tvoří pomocí konstrukce INNER / OUTER JOIN. Pro důkladnou přípravu všech dat v předchozích krocích nebylo nutné další složité upravování během naplňování datové kostky (část zdrojového kódu 5.7).

```sql
INSERT INTO FACT_CEP
SELECT  DPO.IDK_POSKYTOVATEL,
  DP.IDK_PROGRAM,
  DK.IDK_KATEGORIE,
  DR.IDK_ROK,
  GD.ID_KOD          AS KOD,
  GD.NAZEV,
  GD.DATUM_ZAHAJENI,
  GD.DATUM_UKONCENI,
  ... -- neni uplne
  -- 4383==12LET
  FROM (SELECT * FROM GEN_DATA WHERE DATUM_UKONCENI >= SYSDATE - 4383) GD
  LEFT OUTER JOIN DIM_ROK DR
  ON dr.rok = CASE WHEN DR.ROK BETWEEN EXTRACT ( YEAR FROM GD.
    DATUM_ZAHAJENI) AND EXTRACT (YEAR FROM GD.DATUM_UKONCENI)
    THEN DR.ROK ELSE -1 END
  LEFT OUTER JOIN DIM_KATEGORIE DK ON GD.KATEGORIE = DK.KOD
  LEFT OUTER JOIN DIM_PROGRAM DP ON GD.PROGRAM_KOD = DP.KOD
  LEFT OUTER JOIN DIM_POSKYTOVATEL DPO ON GD.POSKYTOVATEL = DPO.KOD
  ... -- neni uplne;
```

Kód 5.7: Plnění faktové tabulky `FACT_CEP` (zdroj vlastní)

Oracle `CUBE`

Další možností, jak vytvořit datovou kostku, je analytická funkce `GROUP BY CUBE`. Výhodu můžeme nalézt v rychlejším sestavení datové kostky. Na druhou stranu není tento přístup moc ohebný a většinou se při tvorbě nové kostky vytvoří také nové volání pomocí `CUBE`[5].

Během použití klauzule `GROUP BY CUBE` je nutno brát ohled na počet vytvořených kombinací. Výsledný počet odpovídá 2^n – viz. ukázka zdrojového kódu 5.8 tzn. pro tři dimenze 2^3 bude celkem osm seskupení). Funkce `AGGREGATE()` v kódu 5.8 představuje agregaci faktové tabulky. [21]

```sql
SELECT c1, c2, c3, aggregate(c4)
FROM table_name
GROUP BY CUBE(c1,c2,c3);
```

Kód 5.8: Ukázka příkazu `GROUP BY CUBE` [21]

`GROUP BY CUBE` ale není vhodný pro tvoření rozsáhlých mnoho dimenzionálních kostek, proto nebyl pro CEP využit.

[5]Podrobnější info na adrese: `https://docs.oracle.com/cd/F49540_01/DOC/server.815/a68003/rollup_c.htm`

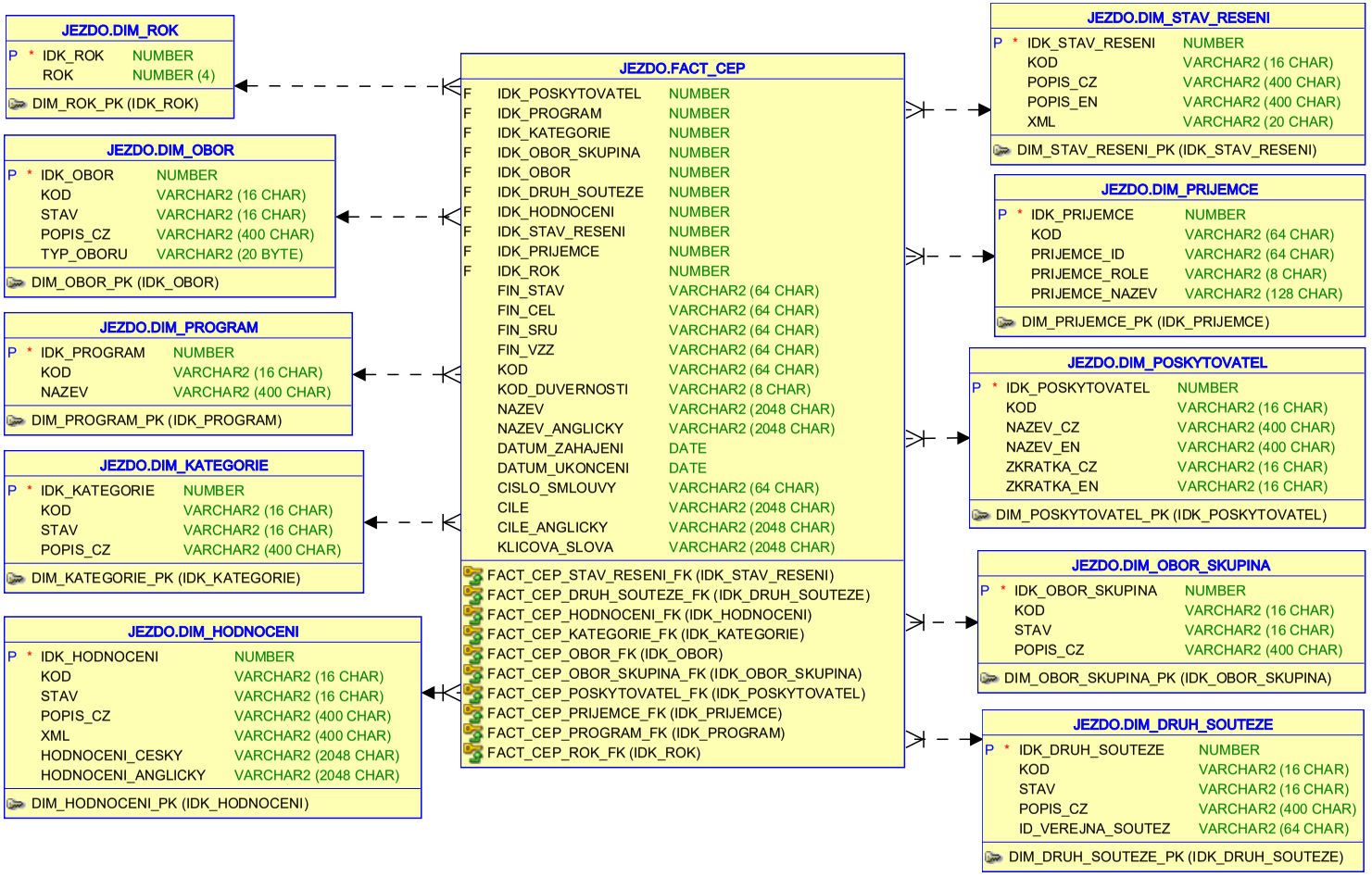
Ukázka výstupu je znázorněna v tabulce 5.3:

ZNACKA	BARVA	CENA
Skoda	Modra	350 000
Skoda	Cervena	340 000
Skoda	NULL	690 000
Volkswagen	Modra	525 000
Volkswagen	Cervena	560 000
Volkswagen	NULL	1 085 888
NULL	Modra	875 000
NULL	Cervena	900 000
NULL	NULL	1 775 000

Tabulka 5.3: Ukázka výstupu `GROUP BY CUBE` [21]

5.5.3 PROCEDURE `CEP_KOSTKA_AKT`

Spuštění procedury zabere přibližně osm minut a je aplikováno na prostředí DW ZČU (Západočeská univerzita v Plzni). Ověření datové pumpy rovněž proběhlo několikanásobným spuštěním skriptu a zkontrolování výsledků napříč daty (data se nezměnila / aktualizace dat nebyla potřeba). Detailní model je zobrazen na obrázku 5.1.



Obrázek 5.1: Model datové kostky `CEP` vygenerovaný z Oracle Developeru (zdroj vlastní)

6 Vizualizace na základě otestovaných dat

V této kapitole bude popsáno navázaní spojení nástrojů Oracle Developer a Power BI. Dále v Power BI budou zobrazeny grafy na základě hotové datové kostky. Ve druhé části budou otestovány výsledky z grafu – zaručeno, že data zůstala zachovalá od počátku stažení dat přes úpravy v DB, tvoření datové kostky až po exportování výsledků do grafu.

6.1 Tvorba analytických dotazů

Analytické dotazy slouží k analýze aktuálních a historických dat. Nástroje BI umožňují práci s velkými daty a jejich následnou prezentaci . Pomocí toho lze odhalit různé korelace, zjištění trendů/poznatků, které se využijí ke strategickým rozhodování. Nyní bude následovat výčet bodů, které podchytávají výhody používání nástrojů BI. [17]

- Vyšší efektivita (rychlost) než u OLTP,
- upozornění na datové anomálie,
- analýzy v reálném čase,
- zjišťování vzorců podle kterých se zákazníci chovají,
- přehled z pohledu historických dat. [17]

6.1.1 Využité technologie

Pro prezentaci dat byly zvažovány 2 vizualizační nástroje – PowerBI a Oracle Apex. PowerBI působí více svěžím a přívětivějším dojmem než Oracle Apex. Obsahuje hodně služeb, které např. slouží pro čistění, transformaci, vizualizaci dat a podporují dotazovací jazyk DAX. Na druhou stranu Oracle Apex je pokročilejší nástroj, který umí spoustu funkcí a má nesporné výhody při využívání Oracle DB jako DW.

Komunita PowerBI je obsáhlá. Data, která dokáže zpracovávat můžou pocházet z různých zdrojů – např. .xlsx, .csv, databázi či cloudových skladů. Power BI se skládá z několika prvků, které navzájem spolupracují. Jedná se

o online službu SaaS (Software as a Service), desktopovou aplikaci Power BI Desktop a mobilní aplikaci – všechny prvky jsou multiplatformní. [18]

Po analýze možností a intuitivnosti ovládání vizualizačních prostředků byl vybrán software power BI.

6.1.2 Import dat do Power BI

Předpokládá se nainstalovaný nástroj **Desktop Power BI**[1]. Pro správnou funkčnost nástrojů Oracle DB a Power BI je zapotřebí mít funkčního klienta **ODAC** (Oracle Data Access Components) ve verzi 11.2 a novější – podrobný návod lze najít na stránkách Microsoftu[2].

Power BI Desktop po spuštění nabízí v horní liště možnost získání dat (Get data). Po výběru příslušného typu DB (Oracle) a vyplnění připojovacích údajů se naváže spojení a připojí se na server. Po navázaní spojení ve schématu, ve kterém je uložená datová kostka, se vybere faktová tabulka `FACT_CEP` a příslušné dimenze, které souvisí s faktovou tabulkou. Po potvrzení se data začnou stahovat – čas stahování závisí na velikosti dat.

Po stažení dat lze v reportovacím okně tvořit vizualizace – jak už dvojrozměrné nebo více rozměrné. Výhoda grafů je, že při správném propojení jednotlivých tabulek se jakákoliv změna grafu propíše do všech ostatních[3].

6.1.3 Dashboard č. 1

Nyní budou popsány dva reprezentující dashboardy vytvořené v Power BI. Dasboard č. 1 (obrázek 6.1) obsahuje celkem sedm vizuálních komponent. Po levé straně jsou zobrazeny tři text-boxy, které obsahují pouze jednu informaci tj. „**Celkový počet projektů**“, „**Počet oboru skupin**“ a „**Počet kategorií**“, jež můžeme považovat za KPI (klíčový ukazatel, key performance indicator).

Vlevo dole se vyskytuje tabulka (anglicky matrix) „**Poskytovatelé**“. V tabulce lze filtrovat jednotlivé poskytovatele po řádkách, roky po sloupcích nebo lze naklikat libovolné hodnoty v tabulce, které se následně propíší do ostatních vizualizací.

Vpravo nahoře se nachází skládaný sloupcový graf (Stacked column chart)

[1] `https://www.microsoft.com/en-us/download/details.aspx?id=58494`

[2] `https://docs.microsoft.com/en-us/power-bi/connect-data/desktop-connect-oracle-database#installing-the-oracle-client`

[3] `https://docs.microsoft.com/en-us/power-bi/visuals/power-bi-report-visualizations`

obsahující informace ohledně „**Projektů přes obory skupin napříč druhem soutěže**“. Jedná se o tří-dimenzionální graf – jednotlivé sloupce (obory skupin), sloupec je rozdělen na tři části (druh soutěže) a samotná data (kódy projektů). Sloupec dat pojmenovaný jako „Blank“ značí, že projekt nebyl zařazen do oboru skupin. Většinou se to stává z důvodu neukončení projektu – stále se činnost vykonává nebo je přerušena.

Koláčový graf (Pie chart), který je umístěn vpravo dole obsahuje informace o „**Stavu řešení**“. Z grafu lze zjistit, v jakém stavu se projekt nachází – odtud vyplývá, že nejvíce projektů je už ukončených.

6.1.4 Dashboard č. 2

Dashboard č. 2 (obrázek 6.2) obsahuje celkem sedm vizuálních komponent. Opět se zde vyskytují dva text-boxy, které obsahují informace „**Celkový počet projektů celkem**“ a „**Finance celkem**“. Finance celkem značí údaj v tisících – např. hodnota 5 značí částku 5 000 českých korun.

Vedle text-boxů se vyskytuje filtr „**Rok**“ ve formě sliceru. Ten má na starost omezení úseku roků.

Pod tím následuje vizualizace ve formě stromové mapy (anglicky Treemap) „**Kategorie přes programy**“, kde se opět jedná o tří-dimenzionální graf – části treemapy označují kategorie, kde každá kategorie obsahuje programy prostřednictvím kódů projektů. V kategorii experimentálním vývoje se nejčastěji vyskytuje program EG (Operační program Podnikání a inovace pro konkurenceschopnost), který se od roku 2015 využívá čím dál častěji (je frekventovanější).

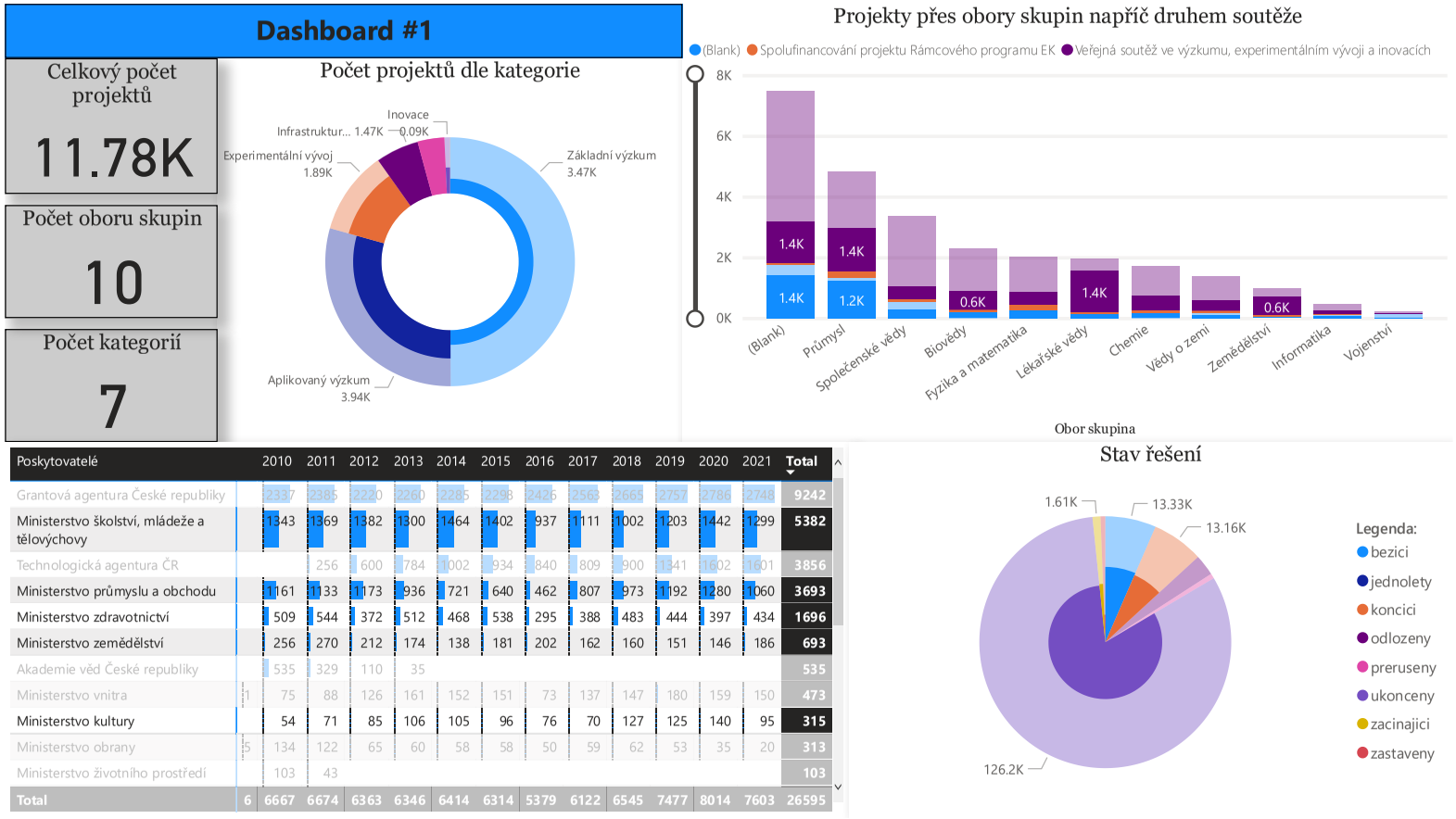
Vlevo dole se nachází řádkový graf (Stacked bar chart) obsahující informace o „**Programech přes finance stavu**“. Zde lze zjistit jaké programy dostaly největší podíl financí.

„**Klíčová slova**“ slouží zde jako experiment. Dalo by se ale zjistit, jaké trendy v jednotlivých projektech jsou populární. Tato vizualizace nebyla v základním balíčku power BI – možnost stáhnutí nejrůznějších vizualizací[4].

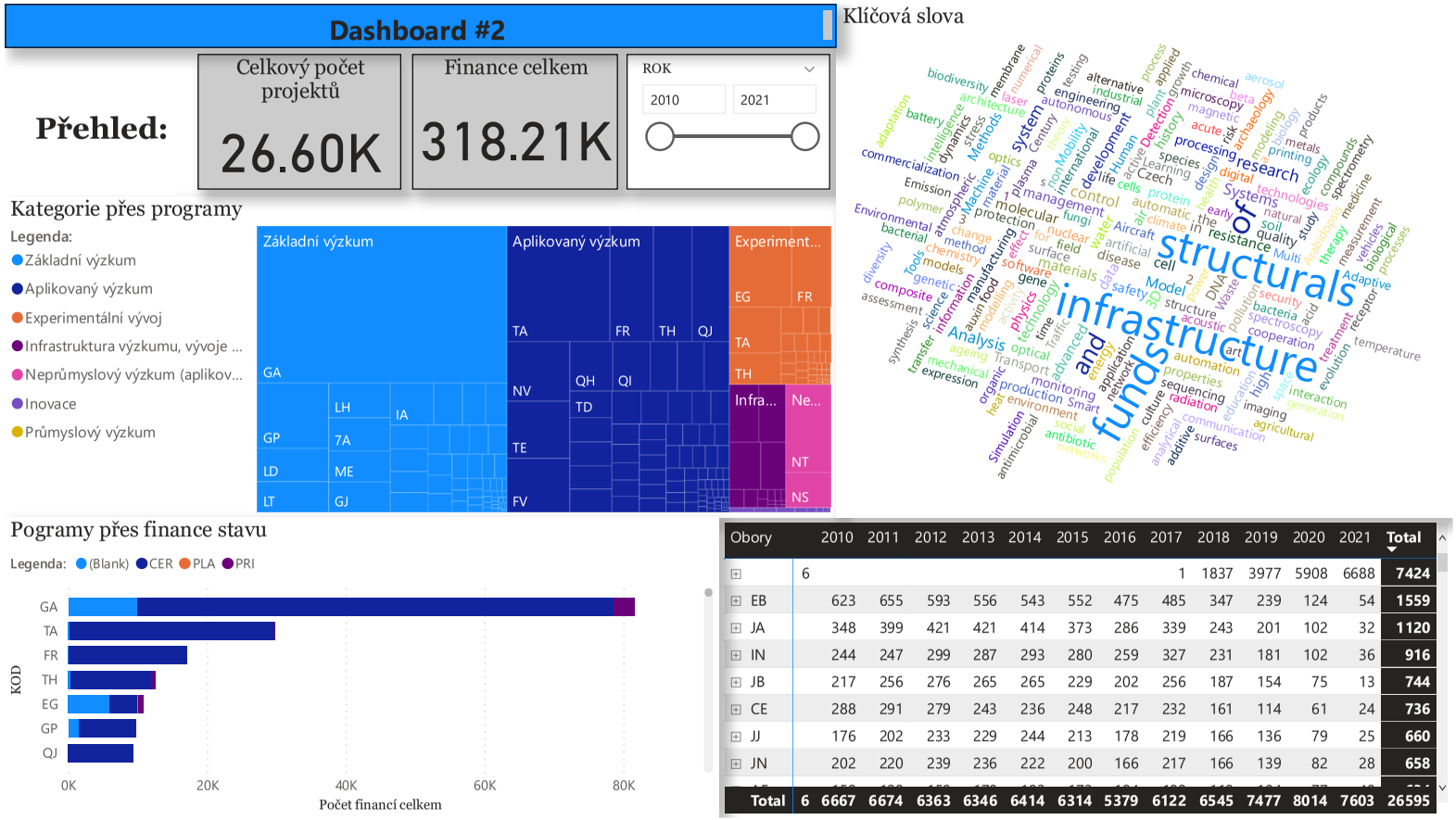
Poslední vizualizace „**Obory**“ je opět ve formě tabulky. Nejpopulárnější existující obor v průběhu let je EB, který se týká Genetiky a molekulární biologie.

Nutno podotknout, že jednotlivé vizualizace v Power BI jdou snadno přizpůsobit požadavkům uživatelů/manažerů.

[4] `https://mindmajix.com/power-bi-visualization-types`



Obrázek 6.1: Dashboard č. 1 – export z nástroje PowerBI (zdroj vlastní)



Obrázek 6.2: Dashboard č. 2 – export z nástroje PowerBI (zdroj vlastní)

6.2 Otestování datové kostky

Testování proběhlo v prostředí Oracle Developeru. V následujících dvou podkapitolách budou otestované některé grafy. Pro úplnost zbytek testů se vyskytuje v ZIP adresáři.

Nutno podotknout, že během testování dat se můžou vyskytnout odlišné výsledky, když se celý proces neprovede v jeden den – tzn. vytvoření datové kostky, nahrání datové kostky do Power BI a otestování v Oracle developeru. Je to z důvodu udržování aktuálnosti v datech. Během těchto procesů se vždy bere v úvahu konstanta 4383, která znamená převod 12ti roků na dny.

6.2.1 Dashboard č. 1

Ověření počtu projektů dle kategorie (zdrojový kód 6.1) – hodnoty napříč Power BI a Oracle databází ze surových dat musí být totožné.

```
SELECT COUNT(DISTINCT ID_KOD)       AS POCET
FROM GEN_DATA
WHERE KATEGORIE = 'ZV' AND DATUM_UKONCENI >= SYSDATE - 4383;
--> Zakladni vyzkum = 13277

SELECT COUNT(DISTINCT ID_KOD)       AS POCET
FROM GEN_DATA
WHERE KATEGORIE = 'AP' AND DATUM_UKONCENI >= SYSDATE - 4383;
--> Aplikovany vyzkum = 7824
```

Kód 6.1: Testování Dashboardu č. 1 vizualizace `Počtu projektů dle kategorie` (zdroj vlastní)

Ověření jestli souhlasí počet projektů přes roky a přes samotné poskytovatele (zdrojový kód 6.2).

```
SELECT COUNT(DISTINCT ID_KOD)       AS POCET
FROM GEN_DATA
WHERE POSKYTOVATEL = 'GAO' AND DATUM_UKONCENI >= SYSDATE - 4383;
--> Grantova agentura Ceske republiky = 9242

SELECT COUNT(DISTINCT ID_KOD)       AS POCET
FROM GEN_DATA
WHERE POSKYTOVATEL = 'MSM'
AND EXTRACT(YEAR FROM TO_DATE(DATUM_UKONCENI, 'DD-MON-RR')) >= 2012
AND EXTRACT(YEAR FROM TO_DATE(DATUM_ZAHAJENI, 'DD-MON-RR')) <= 2012;
--> Ministerstvo skolstvi, mladeze a telovychovy && 2012 = 1382
```

Kód 6.2: Testování Dashboardu č. 1 vizual. `Poskytovatelé` (zdroj vlastní)

Otestování počtu jednotlivých oborů skupin a druhů soutěže přes jednotlivé projekty (zdrojový kód 6.3).

```
SELECT COUNT(DISTINCT ID_KOD)       AS POCET
FROM GEN_DATA
WHERE OBOR_HLAVNI_SKUPINA = 'A' AND DATUM_UKONCENI >= SYSDATE - 4383;
--> Spolecenske vedy = 3350

SELECT COUNT(DISTINCT ID_KOD)       AS POCET
FROM GEN_DATA
WHERE OBOR_HLAVNI_SKUPINA = 'A' AND DRUH_SOUTEZE = 'VS' AND
    DATUM_UKONCENI >= SYSDATE - 4383;
--> Spolecenske vedy && Spolufin. projektu Ramcoveho prog. EK = 2725
```

Kód 6.3: Testování Dashboardu č. 1 vizualizace `Projekty přes obory skupin napříč druhem soutěže` (zdroj vlastní)

6.2.2 Dashboard č. 2

Ověření vizualizace oborů přes roky napříč počtem projektů (zdrojový kód 6.4).

```
SELECT COUNT(DISTINCT ID_KOD)       AS POCET
FROM GEN_DATA
WHERE OBOR_HLAVNI = 'AA' AND DATUM_UKONCENI >= SYSDATE - 4383;
--> Filosofie a nabozenstvi && OBOR_HLAVNI = 188

SELECT COUNT(DISTINCT ID_KOD)       AS POCET
FROM GEN_DATA
WHERE OBOR_VEDLEJSI = 'AA' AND DATUM_UKONCENI >= SYSDATE - 4383;
--> Filosofie a nabozenstvi && OBOR_VEDLEJSI = 30

SELECT COUNT(DISTINCT ID_KOD)       AS POCET
FROM GEN_DATA
WHERE ( OBOR_HLAVNI = 'EB' OR OBOR_VEDLEJSI = 'EB' OR OBOR_VEDLEJSI_DALSI
     = 'EB' ) AND DATUM_UKONCENI >= SYSDATE - 4383;
--> Filosofie a nabozenstvi && OBOR_HLAVNI && OBOR_VEDLEJSI &&
    OBOR_VEDLEJSI_DALSI = 1559
```

Kód 6.4: Testování Dashboardu č. 2 vizualizace `Počet projektů přes obory` (zdroj vlastní)

7 Závěr

Hlavním cílem práce bylo vytvořit datovou kostku. Během tvoření datové kostky bylo nutné v teoretické části práce probrat jednotlivé koncepty DW, DB a struktury CEP. V rámci implementační části byla vybrána vhodná data pro reprezentaci datové kostky. V dalším kroku byla jednotlivá data formátu JSON importována do RDBS Oracle. Bylo nutné rozčlenit a určit data, která nás budou zajímat v rámci DW ZČU. Před naplněním datové kostky bylo nutné data očistit a připravit k naplnění. V závěru práce bylo nutné otestovat datovou kostku pomocí nahrání do analytického nástroje Power BI, kde se vytvořily grafické výstupy. Otestování výstupů probíhalo napříč daty, která byla nahrána do DB v původním formátu (a jejich obsah nebyl jakkoli pozměněn).

Nedílnou součásti práce jsou vytvořené skripty a data obsažená v ZIP adresáři. Skripty jsou patřičně rozděleny podle toho, zda se jedná o plnící, pomocné, nebo jsou obsahem testování. Obsah dat se dělí na dvě části. První soubor obsahuje stažená data z API ve formátu JSON. Druhý soubor obsahuje vyexportována data z datové kostky ve formátu CSV, který se dá použít nezávislé na Oracle DB.

Majoritní část znalostí byla čerpána z knížek Datové sklady Agilní metody a business intelligence od Roberta Laberga [14] a knížky Warehouse Toolking The Definitive Guide to Dimensional Modeling od Raplha Kimballa a Margyho Rosse [25]. Ostatních zdroje byly čerpány z online publikací.

Do budoucna by bylo vhodné provést zautomatizování celého procesu. To by znamenalo vytvořit proces/nástroj, který by stahoval předem určené JSONy z API a následně pomocí jobu[1] spouštět procedury sloužící k aktualizaci dat. Dalším možným rozšířením by bylo napojení dat na stávající systémy pro evidenci projektů na ZČU a korekce správnost záznamu v GaP (Granty a Projekty). V neposlední řadě by šlo provést hledání hlubších analýz nad zpracovanými daty, které by vedly k nalezení zajímavých korelací.

[1] `https://docs.oracle.com/cd/E18283_01/server.112/e17120/scheduse002.htm`

Seznam použitých zkratek

ACID – **A**tomicity, **C**onsistency, **I**solation, **D**urability

ADW – **A**utonomous **D**ata **W**arehouse

API – **A**pplication **P**rogramming **I**nterface

ARIMA – **A**uto**R**egressive **I**ntegrated **M**oving **A**verage

BCNF – **B**oyce-**C**oddova **N**ormální **F**orma

BI – **B**usiness **I**ntelligence

CEA – **C**entrální **E**vidence **A**ktivit VaVaI

CEP – **C**entrální **E**vidence **P**rojektů

CIV – **C**entrum **I**nformatizace a **V**ýpočetní techniky

CLOB – **C**haracter **L**arge **OB**ject

CRM – **C**ustomer **R**elationship **M**anagement

DB – **D**ata**b**áze

DBMS – **D**ata**b**ase **M**anagement **S**ystem

DBS – **D**ata**b**ázový **S**ystém

DW – **D**ata **W**arehouse – datový sklad

ETL – **E**xtract, **T**ransform and **L**oad

ELT – **E**xtract, **L**oad and **T**ransform

FK – **F**oreign **K**ey

GaP – **G**ranty **a P**rojekty

IBM – **I**nternational **B**usiness **M**achines Corporation

IS – **I**nformační **S**ystém

IS VaVaI – **I**nformační **S**ystém **V**ýzkumu, **V**ývoje a **I**novací

JSON – **J**ava**S**cript **O**bject **N**otation

KDD – **K**nowledge **D**iscovery in **D**atabases

KPI – **K**ey **P**erformance **I**ndicator

NF – **N**ormální **F**ormy

NS – **N**euronové **S**ítě

ODAC – **O**racle **D**ata **A**ccess **C**omponents

OLAP – **O**n**l**ine **A**nalytical **P**rocessing

OLTP – **O**n**l**ine **T**ransaction **P**rocessing

ORDB – **O**bjektově **R**elační **D**atab**á**ze

PJNF – **P**roject-**J**oin **N**ormal **F**orm

PK – **P**rimary **K**ey

PL/SQL – **P**rocedural **L**anguage/**S**tructured **Q**uery **L**anguage

RDB – **R**elační **D**atab**á**ze

RDBMS – **R**elational **D**atab**a**se **M**anagement **S**ystem

REST – **R**epresentational **S**tate **T**ransfer

RIV – **R**ejstřík **I**nformací o **V**ýsledcích

SEMMA – **S**ample **E**xplore **M**odify **M**odel **A**ssess

SOAP – **S**imple **O**bject **A**ccess **P**rotocol

SŘBD – **S**ystém **Ř**ízení **B**áze **D**at

SQL – **S**tructured **Q**uery **L**anguage

ÚV ČR – **Ú**řad **V**lády **Č**eské **R**epubliky

VES – **E**vidence **V**eřejných **S**outěží ve VaVaI

ZČU – **Z**ápadočeská **U**niverzita v **P**lzni

Literatura

[1] Czarski, C. *Report and Form on a REST Service: Low Code with APEX 19.1* [online]. https://blogs.oracle.com/, 2019. [cit. 2022/04/24]. Dostupné z: https://blogs.oracle.com/apex/post/report-and-form-on-a-rest-service-low-code-with-apex-191#smaller-text.

[2] DaCosta, V. *20 Best ETL Tools in 2022* [online]. https://hevodata.com, 2021. [cit. 2022/01/26]. Dostupné z: https://hevodata.com/blog/best-etl-tools-data-warehouse/.

[3] Debnath, M. *How do OODBMS and ORDBMS Differ from RDBMS?* [online]. https://www.databasejournal.com/, 2020. [cit. 2022/04/23]. Dostupné z: https://www.databasejournal.com/ms-access/how-do-oodbms-and-ordbms-differ-from-rdbms/.

[4] Education, I. C. *Data Mining* [online]. https://www.ibm.com, 2021. [cit. 2022/04/18]. Dostupné z: https://www.ibm.com/cloud/learn/data-mining#toc-what-is-da-4ZYZUvj4.

[5] Foote, K. D. *A Brief History of Data Modeling* [online]. https://www.dataversity.net/, 2017. [cit. 2022/03/12]. Dostupné z: https://www.dataversity.net/brief-history-data-modeling/.

[6] Gittlenová, S. *Jak datová jezera zlepšují analýzu dat* [online]. https://www.computerworld.cz/, 2017. [cit. 2022/04/15]. Dostupné z: https://www.computerworld.cz/clanky/jak-datova-jezera-zlepsuji-analyzu-dat/.

[7] Holdaway, K. *Predictive Analytics: Development and Deployment of Upstream Data Driven Models* [online]. https://www.researchgate.net/, 2012. [cit. 2022/04/19]. Dostupné z: https://www.researchgate.net/publication/254536111_Predictive_Analytics_Development_and_Deployment_of_Upstream_Data_Driven_Models.

[8] Hrychová, T. *Systém řízení báze dat* [online]. https://wikisofia.cz/, 2017. [cit. 2022/01/25]. Dostupné z: https://wikisofia.cz/wiki/Systém_řízení_báze_dat.

[9] IBM. *Relational Database* [online]. https://www.ibm.com/, 2022. [cit. 2022/03/20]. Dostupné z: https://www.ibm.com/ibm/history/ibm100/us/en/icons/reldb/.

[10] IBM. *IBM SPSS software* [online]. https://www.ibm.com/, 2022. [cit. 2022/04/19]. Dostupné z: https://www.ibm.com/analytics/spss-statistics-software.

[11] Ing. Roman Danel, P. *Dolování dat* [online]. https://homel.vsb.cz/, 2010. [cit. 2022/04/19]. Dostupné z: https://homel.vsb.cz/~dan11/is_skripta/IS%202010%20-%20Danel%20-%20Dolovani%20dat.pdf.

[12] Javatpoint. *Fourth normal form (4NF)* [online]. https://www.javatpoint.com/, 2022. [cit. 2022/04/28]. Dostupné z: https://www.javatpoint.com/dbms-forth-normal-form.

[13] Javatpoint. *Fifth normal form (5NF)* [online]. https://www.javatpoint.com/, 2022. [cit. 2022/04/28]. Dostupné z: https://www.javatpoint.com/dbms-fifth-normal-form.

[14] Laberge, R. *Datové sklady, Agilní metody a business intelligence.* Computer Press v Brně ve společnosti Albatros Media a. s., 2011. ISBN 978-80-251-3729-1.

[15] Lucidchart. *What is a Database Model* [online]. https://www.lucidchart.com/, 2022. [cit. 2022/04/23]. Dostupné z: https://www.lucidchart.com/pages/database-diagram/database-models.

[16] Matillion. *The Types of Databases (with Examples)* [online]. https://www.matillion.com/, 2020. [cit. 2022/03/20]. Dostupné z: https://www.matillion.com/resources/blog/the-types-of-databases-with-examples.

[17] Microsoft. *What is business intelligence?* [online]. https://powerbi.microsoft.com/, 2022. [cit. 2022/04/24]. Dostupné z: https://powerbi.microsoft.com/en-au/what-is-business-intelligence/.

[18] Microsoft. *What is Power BI?* [online]. https://docs.microsoft.com/, 2022. [cit. 2022/04/24]. Dostupné z: https://docs.microsoft.com/en-us/power-bi/fundamentals/power-bi-overview.

[19] Oracle. *What is a Relational Database (RDBMS)?* [online]. https://www.oracle.com/, 2021. [cit. 2022/01/26]. Dostupné z: https://www.oracle.com/database/what-is-a-relational-database/.

[20] Oracle. *Autonomous Database for analytics and data warehousing (ADW)* [online]. https://www.oracle.com, 2022. [cit. 2022/04/18]. Dostupné z: https://www.oracle.com/autonomous-database/autonomous-data-warehouse/?bcid=6030055588001.

[21] Oracle. *Oracle CUBE* [online]. https://www.oracletutorial.com/, 2022. [cit. 2022/04/24]. Dostupné z: https://www.oracletutorial.com/oracle-basics/oracle-cube/.

[22] Oracle. *JSON in Oracle Database* [online]. https://docs.oracle.com/, 2022. [cit. 2022/04/23]. Dostupné z: https://docs.oracle.com/database/121/ADXDB/json.htm.

[23] Oracle. *BFILENAME* [online]. https://docs.oracle.com/, 2022. [cit. 2022/04/23]. Dostupné z: https://docs.oracle.com/cd/B19306_01/server.102/b14200/functions012.htm.

[24] Oracle. *What is OLTP?* [online]. https://www.oracle.com, 2022. [cit. 2022/04/18]. Dostupné z: https://www.oracle.com/database/what-is-oltp/.

[25] Ralph Kimball, M. R. *Warehouse Toolkit*. John Wiley and Sons, Inc., Indianapolis, Indiana, 2013. ISBN 978-1-118-53080-1.

[26] Růžičková, M. *Data mining — Co? Jak? K čemu?* [online]. https://medium.com/, 2018. [cit. 2022/04/19]. Dostupné z: https://medium.com/edtech-kisk/data-mining-co-jak-k-%C4%8Demu-c5176179303b.

[27] Saxon, C. *How to Store, Query, and Create JSON Documents in Oracle Database* [online]. https://blogs.oracle.com/, 2021. [cit. 2022/04/23]. Dostupné z: https://blogs.oracle.com/sql/post/how-to-store-query-and-create-json-documents-in-oracle-database.

[28] Sinha, T. *OLAP vs. OLTP: What's the Difference?* [online]. https://www.ibm.com, 2021. [cit. 2022/04/18]. Dostupné z: https://www.ibm.com/cloud/blog/olap-vs-oltp.

[29] Stitch. *What is a Data Lake? Examples and Solutions [Free Guide]* [online]. https://www.stitchdata.com/, 2021. [cit. 2022/04/15]. Dostupné z: https://www.stitchdata.com/resources/what-is-data-lake.

[30] Stitch. *OLTP and OLAP: a practical comparison* [online]. https://www.stitchdata.com/, 2022. [cit. 2022/04/23]. Dostupné z: https://www.stitchdata.com/resources/oltp-vs-olap.

[31] Talend. *What is a Data Mart?* [online]. https://www.talend.com, 2021. [cit. 2022/01/26]. Dostupné z: https://www.talend.com/resources/what-is-data-mart/.

[32] Talend. *What is ETL?* [online]. https://www.talend.com, 2021. [cit. 2022/01/26]. Dostupné z: https://www.talend.com/resources/what-is-etl/.

[33] Taylor, D. *Star and Snowflake Schema in Data Warehouse with Model Examples* [online]. https://www.guru99.com, 2021. [cit. 2022/01/26]. Dostupné z: https://www.guru99.com/star-snowflake-data-warehousing.html.

[34] ThinkAutomation. *The history of databases* [online]. https://www.thinkautomation.com/, 2022. [cit. 2022/03/20]. Dostupné z: https://www.thinkautomation.com/histories/the-history-of-databases/.

[35] VaVaI, I. *O IS VAVAI* [online]. https://www.isvavai.cz/, 2022. [cit. 2022/04/19]. Dostupné z: https://www.isvavai.cz/is?s=o-is-vavai&fbclid=IwAR27MFpgrHC1Jq6ayHtS4V2_JSkDT0IHqiH6E-dJx-yT3VHUCUVXqKImR9c.

[36] Vovcr. *Potřeba a význam databázových systémů* [online]. https://www.vovcr.cz/, 2022. [cit. 2022/01/26]. Dostupné z: https://www.vovcr.cz/odz/tech/393/page01.html.

[37] Zain Aftab, K. M. A. F. B. W. I. – Abdullah, M. *Automatic NoSQL to Relational Database Transformation with Dynamic Schema Mapping* [online]. https://downloads.hindawi.com, 2020. [cit. 2022/04/21]. Dostupné z: https://downloads.hindawi.com/journals/sp/2020/8813350.pdf.

[38] Úřad_vlády_České_republiky. *Specifikace API pro IS VaVaI 3.0* [online]. https://api.isvavai.cz/, 2021. [cit. 2022/04/23]. Dostupné z: https://api.isvavai.cz/dokumenty/SpecifikaceApiVavai3_v2.1.0_211221.pdf.

Přílohy

A Uživatelská příručka

Příloha obsahuje stručný postup pro stažení dat z API, nahrání dat do DB a spuštění datové kostky v Power BI. Jednotlivé kroky na sebe navazují. Při případném vzniklém problému během jednotlivých kroků obsahuje ZIP adresář už předem stažená data z API a také vyexportována data z datové kostky. Tzn. v případě nedosažení chronologického postupu do zdárného konce lze krok přeskočit a pokračovat v dalším kroku, ale bez aktuálních dat.

A.1 Stažení dat z CEPu

Pro stažení dat z CEPu prostřednictvím API je vhodné mít nainstalovaný nástroj, který usnadňuje práci s API. Doporučením je nástroj Insomnia, který se používá v této práci. Pro přístup k datům je nutné mít platný token, který má přístup k API do CEPu IS VaVaI.

Předpokládá se nastudovaná specifikace API. Následně je vhodné si token provázat v Insomnii z důvodu redundance tokenu. Nakonec pomocí metody POST provoláme API s příslušnými vyplněnými parametry. Stažená data si uložíme do adresáře, se kterými budeme nadále pracovat.

A.2 Nahrání dat do databáze

V případě, kdy nemáme k dispozici data z kroku č. 1, tak lze použít data obsažena v adresáři `Vstupni_data\export_cep`.

V této části se budou používat nástroj Oracle Developer. Cílem první části je vytvoření directory, které bude odkazovat do adresáře s nahranými daty z CEPu. Po dosažení těchto výsledků je vhodné požádat oprávněnou osobu, aby Vám přidělila práva nebo ji požádat o nahrání dat na server do svého adresáře. Po úspěšném nahrání dat na server a vytvoření directory lze spouštět skripty. Skripty se nachází v adresáři `Aplikace_a_knihovny`, kde jsou jednotlivé části členěné do složek. Při nahrávání dat do DB je postačující využít složky `CISELNIKY` a `DETAIL`, kde jsou skripty členěné do jednotlivých souborů nebo do jednoho velkého souboru, který obstarává hromadné vytvoření tabulek a nahrání dat do DB.

Datová kostka je vytvářena stejným způsobem, jako předešlé skripty – skript

v adresáři `Aplikace_a_knihovny\PACKAGE` a spuštění package pomocí příkazu `EXEC PG_CEP.CEP_KOSTKA_AKT`. Tím se zajistí vytvoření tabulek a naplnění dat (data jsou nová nebo aktualizovaná).

A.3 Nahrání datové kostky do Power BI

Předpokladem jsou stažené nástroje Power BI a klient ODAC. Klient ODAC je zapotřebí, jestliže se budou data nahrávat z DB do Power BI.

Když data v DB nemáme z jakéhokoliv důvodu, je možnost stáhnout data ze složky `Vstupni_data\export_datove_kostky`, kde jsou ve formátu CSV. Data postačí nahrát do Power BI a určit závislosti mezi jednotlivými tabulkami – faktová tabulka se propojí s každou dimenzí pomocí IDK.

Po nahrání nebo propojení datové kostky a Power BI lze vytvářet jednotlivé dashboardy, pomocí kterých lze vytvářet různé analýzy a zjišťovat nečekané korelace.

A.4 Testování dashboardů

Předpokládá se, že předešlé kroky byly úspěšně splněny. Testování probíhá v nástroji Oracle Developer, kde se jedná o ověření pravdivosti údajů z grafů. Jednotlivé skripty se nachází v adresáři `Aplikace_a_knihovny\TESTOVANI`. Výsledky testů souhlasí s dashboardem č. 1 a dashboardem č. 2.

B Obsah ZIP souboru

K práci je přiložen ZIP adresář, který obsahuje následující strukturu.

Adresářová struktura

- `Text_prace` ..... obsah zdrojových kódů
- `Aplikace_a_knihovny` ..... scripty, package, testování
  - `oracle_scripty` .....
    - `CISELNIKY` .....
    - `DETAIL` .....
    - `DIM` .....
    - `FACT` .....
    - `OTHERS` .....
    - `PACKAGE` .....
    - `TESTOVANI` .....
    - `VIEW` .....
  - `powerbi-dashboard.pbix` .....
  - `insomnia-configure.json` .....
- `Vstupni_data` ..... export dat z API + export datové kostky
  - `export_cep` .....
    - `filtr_detail` .....
    - `ciselniky` .....
    - `SpecifikaceApi.pdf - specifikace API v CEPu` .....
  - `export_datove_kostky` .....
- `Vysledky` ..... export dashboardu z Power BI
  - `powerbi-export-dashboard.pdf` .....
- `Readme.txt` ..... detailní popis struktury adresáře