

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

Diplomová práce

Tvorba rozsáhlých úložišť patentových dat

Místo této strany bude
zadání práce.

Prohlášení

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

V diplomové práci jsou použity názvy programových produktů, firem apod., které mohou být ochrannými známkami nebo registrovanými ochrannými známkami příslušných vlastníků.

V Plzni dne 26. dubna 2022

Bc. Vojtěch Danišík

Poděkování

Děkuji panu Doc. Ing. Daliboru Fialovi, Ph.D. za ochotu při vedení diplomové práce a rady s jejím vypracováním.

Abstract

Creation of large-scale patent data repositories. The aim of the diploma thesis is to get acquainted with the available sources of patent data and to create extensive local repositories of patent data enabling their effective searching and mining. The first part of the thesis thoroughly describes the types of patents, existing data sources and file formats in which patents are stored. Subsequently, the applicable technologies for searching and mining are described. The second part of the thesis is devoted to the selection of usable data and the implementation of selected technologies. Several queries and scenarios have been created to test efficient mining. The results of the testing are part of this work.

Abstrakt

Cílem diplomové práce je seznámit se s dostupnými zdroji dat o patentech a vytvořit rozsáhlá lokální úložiště patentových dat umožňující jejich efektivní prohledávání a vytěžování. První část práce důkladně popisuje typy patentů, existující zdroje dat a formáty souborů, ve kterých se patenty ukládají. Následně jsou popsány použitelné technologie pro prohledávání a vytěžování. Druhá část práce se věnuje výběru použitelných dat a implementaci vybraných technologií. Pro otestování efektivního vytěžování bylo vytvořeno několik dotazů a scénářů. Výsledky testování jsou součástí této práce.

Obsah

1	Úvod	1
2	Patent	2
2.1	Patent vs Užitný vzor	2
2.2	Patent vs Průmyslový vzor	2
2.3	Patent vs Ochranná známka	2
2.4	Patent vs Autorská práva	2
3	Databáze	3
3.1	Systém řízení báze dat	3
3.2	Komponenty databáze	4
3.3	Typy databází	4
3.3.1	Relační databáze	4
3.3.2	Objektově-orientovaná databáze	6
3.3.3	NoSQL databáze	7
3.3.4	Databáze Klíč-Hodnota	7
3.3.5	Grafová databáze	8
3.3.6	Dokumentová databáze	9
3.4	Existující řešení	10
3.4.1	MySQL	10
3.4.2	PostgreSQL	10
3.4.3	ConceptBase	11
3.4.4	LevelDB	11
3.4.5	Redis	11
3.4.6	MongoDB	11
3.4.7	Elasticsearch	11
3.4.8	Neo4j	11
3.5	Jazyky	11
3.5.1	Structured Query Language	13
3.5.2	MongoDB Query Language	14
3.5.3	Cypher Query Language	15
4	Návrh úložiště	17
4.1	Výběr patentů	17
4.1.1	Zdroje dat	17
4.1.2	Atributy	20

4.1.3	Závěr průzkumu	22
4.2	Výběr databáze	22
4.2.1	Výběr typu databáze	22
4.2.2	Výběr z existujících řešení	22
4.2.3	Závěr průzkumu	22
5	Implementace úložiště	23
5.1	Adresářová struktura	23
5.1.1	Patenty	23
5.1.2	Docker	23
5.2	Implementace databáze	23
5.2.1	MySQL	23
5.2.2	MongoDB	23
5.3	Výsledné úložiště	23
5.3.1	Technologické požadavky	23
5.3.2	Docker	23
5.3.3	Inicializace MySQL	23
5.3.4	Inicializace MongoDB	23
6	Rozšiřitelnost úložiště	24
6.1	Přidávání nových patentů	24
6.2	Zjišťování autorů pro české patenty	24
6.3	Automatické stahování dat z ověřených zdrojů	25
7	Ověření efektivního vytěžování	27
7.1	Mongo + Elasticsearch	27
7.2	MySQL	27
7.2.1	Scénář č.1	27
7.2.2	Scénář č.2	28
7.2.3	Scénář č.3	29
7.2.4	Scénář č.4	29
7.2.5	Scénář č.5	30
7.2.6	Scénář č.6	31
7.2.7	Scénář č.7	32
7.2.8	Scénář č.8	33
7.2.9	Scénář č.9	33
8	Závěr	35
	Zkratky	36

Literatura	37
A Uživatelská dokumentace	38
B Vzhled modulů	39

1 Úvod

Test

Cílem této práce je se seznámit s dostupnými zdroji dat o patentech a vytvořit rozsáhlá lokální úložiště patentových dat umožňující jejich efektivní vytěžování. Zdroje dat musí poskytovat své databáze patentů (žádosti i publikace) zdarma a patenty musí obsahovat předem stanovené povinné atributy, aby je bylo možné použít. Získaná data budou následně uložena do specifického typu databáze, která bude umožňovat co nejefektivnější vytěžování uložených dat. To znamená rychlé vyhledávání správných výsledků v relativně krátkém čase pro miliony (až desítky milionů) záznamů. Import dat bude řešen pomocí jednoduché aplikace, která bude procházet všechna data a filtrovat ty patenty, kterým chybí některé povinné údaje (i přes to, že struktura obsahuje elementy, ve kterých se údaj má nacházet), nebo jsou nevalidní.

1-2 odstavce
-> až se udělá kapitola patent - pro získání více info

2 Patent

todo

základní informace + uložště dat + info o zdrojích (patentové úřady atp)

2.1 Patent vs Užitný vzor

2.2 Patent vs Průmyslový vzor

2.3 Patent vs Ochranná známka

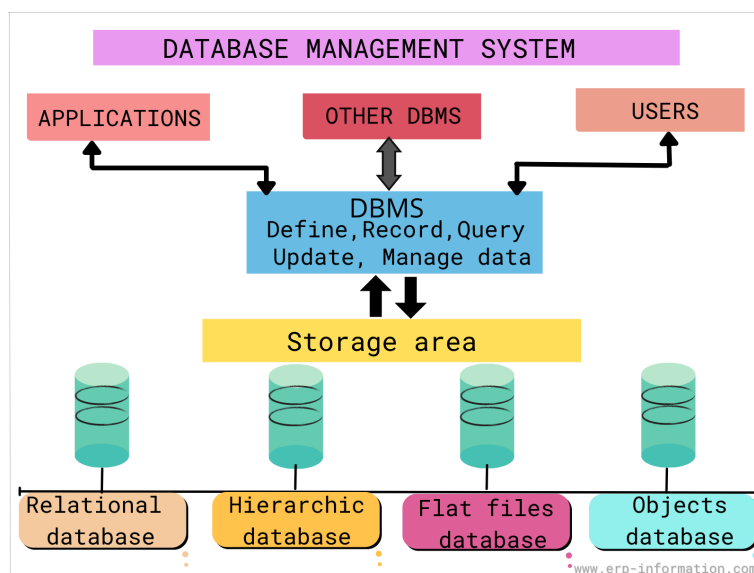
2.4 Patent vs Autorská práva

3 Databáze

Termín databáze označuje organizovanou kolekci strukturovaných informací nebo dat, která jsou typicky ukládána elektronicky v počítačovém systému. Data / informace lze nazvat jako fakta vztahující se k libovolnému uvažovanému objektu. Typický příklad objektu je člověk, jehož fakta jsou: jméno, věk, výška, váha a mnoho dalších [9].

3.1 Systém řízení báze dat

Pro správu dat v databázi a její řízení je potřeba komplexní software, který se nazývá **Systém Řízení Báze Dat** (SŘBD, anglicky DBMS). SŘBD slouží jako interface mezi samotnou databází a koncovým uživatelem (může být i program), umožňující jak vytěžování a aktualizaci dat, tak i možnosti nastavení záloh a jiných administrativních operací [1]. V dnešním světě existuje několik různých DBMS (například Relační DBMS, Objektově-orientované DBMS).



Obrázek 3.1: Systém řízení báze dat

3.2 Komponenty databáze

Všechny databáze sestávají z pěti základních komponent, nehledě na použitý typ databáze [8, 9]:

- **Hardware** - Fyzické stroje (počítače, servery, pevné disky, ...) na kterých běží databázový software.
- **Software** - Databázový software poskytuje uživateli / programu kontrolu nad databází. Zahrnuje to samotný databázový software, operační systém, software pro správu sdílení dat mezi uživateli a programy pro přístup k datům v databázi.
- **Data** - Nezpracované a neorganizované fakty, které je potřeba zpracovat. Administrátor databáze organizuje tyto data a dává jim význam. Data se obecně skládají hlavně z faktů, observací, percepce, čísel, znaků a mnoho dalších.
- **Jazyk** - Typický příklad použití jazyku je přístup k datům, přidávání nových dat, úpravu již existujících dat z databáze. Uživatel / program napíše specifické příkazy v jazyku pro přístup k datům (Database Access Language) a tyto příkazy následně pošle databázi ke zpracování. Více viz kapitola č. 3.5.
- **Procedury** - Procedura obsahuje předpřipravený seznam příkazů, které se následně vykonávají po zavolání dané procedury.

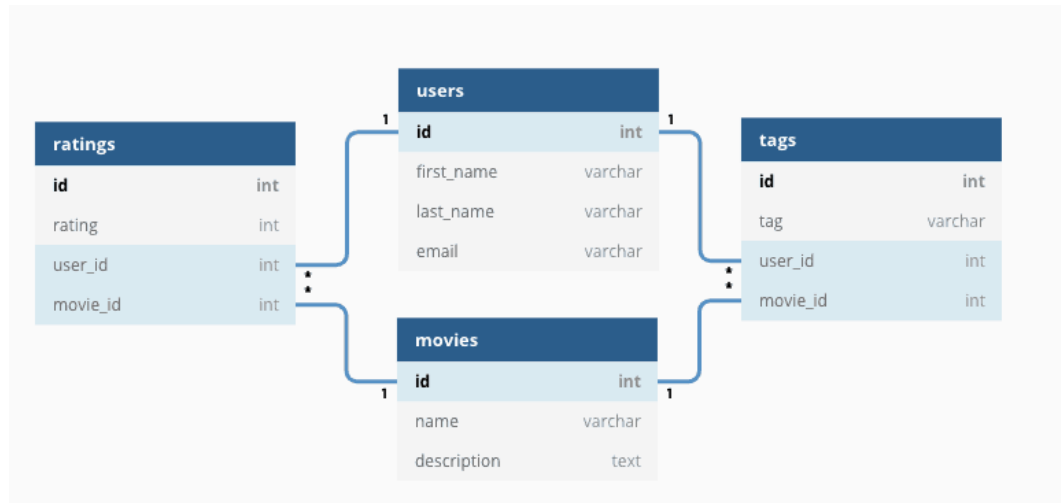
3.3 Typy databází

V dnešním světě existuje mnoho různých typů databází. Výběr nejlepšího typu databáze pro konkrétní organizaci závisí na tom, jak organizace zamýšlí data používat. V této kapitole je vypsáno pouze pár typů, protože vznikají stále nové, méně známé typy databází, které jsou tvořeny pro specifické požadavky (například finanční, vědecké) [1, 4].

3.3.1 Relační databáze

Název relační databáze pochází ze způsobu, jakým jsou data uložena, a to ve více souvisejících tabulkách. Data v tabulkách jsou uložena v řádcích a sloupcích. Relační databáze jsou velice spolehlivé a podporují všechny čtyři žádoucí vlastnosti databázových transakcí ACID. Pro co nejeфекtivnější využití tohoto typu databáze je potřeba ukládat pouze dobře strukturovaná

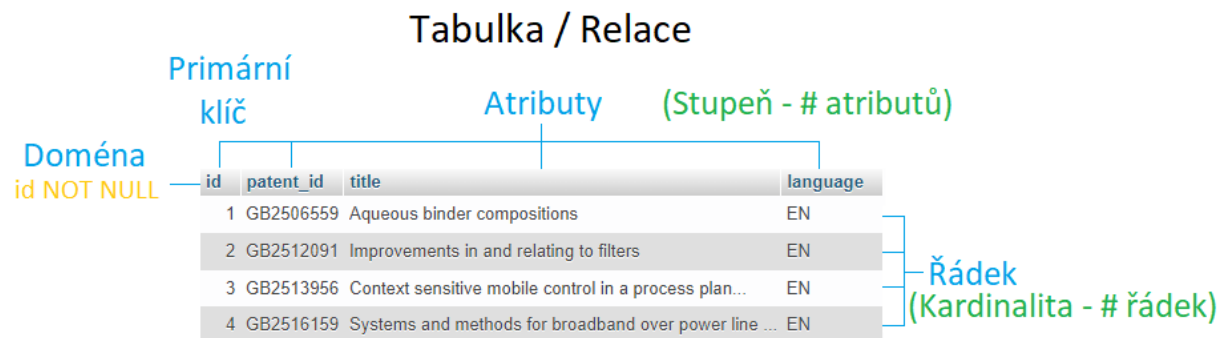
data, pro částečně strukturovaná či nestrukturovaná data je vhodné použít například grafové nebo dokumentově založené databáze. Typické relační databáze jsou například: Microsoft SQL Server, Oracle Database, MySQL, PostgreSQL. Ukázkou relační databáze lze vidět na obrázku č. 3.2.



Obrázek 3.2: Ukáзка relační databáze

Datový model

Relační datový model obsahuje několik fundamentálních konceptů. Koncepty lze vidět na obrázku č. 3.3.



Obrázek 3.3: Koncepty relačního datového modelu.

První z konceptů se nazývá **Relace**, což je dvou-dimenzionální tabulka, která se používá pro ukládání kolekce datových elementů. Tabulka je tvořena řádky a sloupce, kde řádky reprezentují záznamy a sloupce reprezentují

atributy.

Řádka je další koncept relačního modelu, která pouze reprezentuje jeden záznam v tabulce.

Další z konceptů je **Atribut**, který reprezentuje sloupec v tabulce, neboli vlastnosti jednotlivých řádků (například jméno, příjmení, věk, ...).

Koncept **Doména atributů** slouží k definici vlastností pro každou hodnotu daného atributu. Pomocí domény lze určit, zda hodnoty daného atributu mohou být prázdné, budou dlouhé maximálně 50 znaků nebo například určit datový typ atributu (textová hodnota, číslo, ...).

Další z konceptů je **Stupeň**, který pouze určuje počet atributů v dané relaci.

Kardinalita určuje počet řádků / záznamů existujících v dané relaci.

Koncept **Relační schéma** popisuje návrh a strukturu relace. Obsahuje názvy tabulek, jejich atributy a typy atributů. Relační schéma pro naši tabulku lze vidět na obrázku č. 3.4.

```
PATENTS(id INT NOT NULL,  
        patent_id VARCHAR(50),  
        title VARCHAR(300),  
        language VARCHAR(2)  
        )
```

Obrázek 3.4: Relační schéma pro tabulku na obrázku č. 3.3.

Relační instance reprezentuje kolekci záznamů, které jsou uloženy v tabulce v určitém čase.

Poslední koncept **Relační klíč** je atribut / seznam atributů, které lze využít jako unikátní identifikátor jedné entity v tabulce, případně k určení vazby mezi dvěma relacemi. Existuje šest typů relačních klíčů - kandidátní, super, složený, primární, cizí, sekundární / alternativní [7].

Výhody

todo

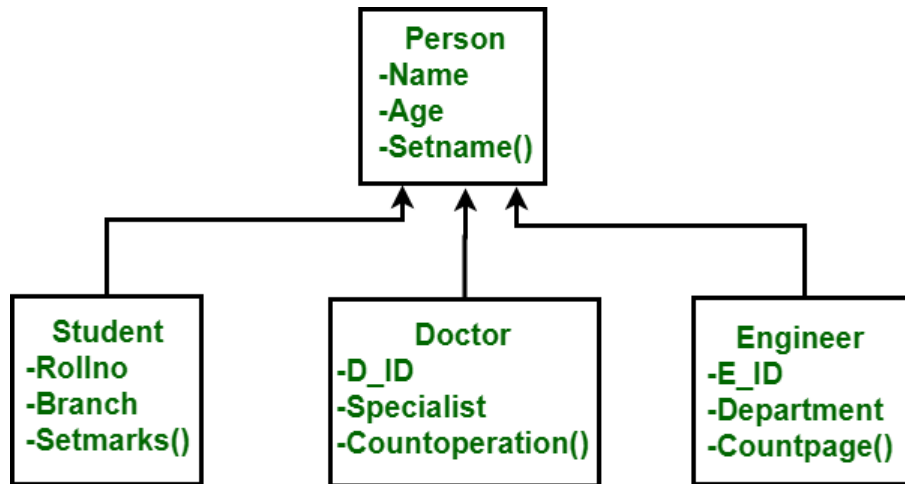
Nevýhody

todo

3.3.2 Objektově-orientovaná databáze

Objektově-orientovaná databáze je založena na objektově-orientovaném programování, kdy data a všechny jejich atributy a metody jsou svázány dohromady jako objekt. Stejně jako relační databáze, i objektově-orientované

databáze odpovídají standardům ACID. Typické příklady jsou například: ObjectStore, ConceptBase. Ukázku objektově-orientované databáze lze vidět na obrázku č. 3.5.



Obrázek 3.5: Ukázka objektově-orientované databáze

Datový model

todo

Výhody

todo

Nevýhody

todo

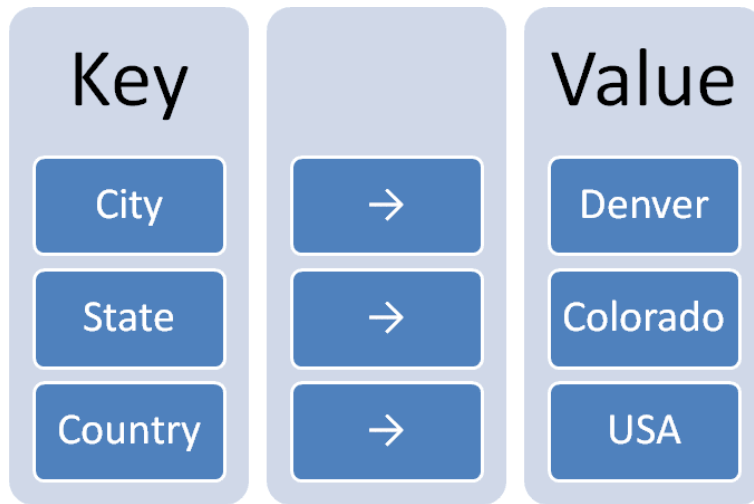
3.3.3 NoSQL databáze

NoSQL je široká kategorie databází, které nepoužívají SQL jako svůj primární jazyk pro přístup k datům. Tyto typy databází jsou také někdy označovány jako nerelační databáze. V NoSQL databázích se pracuje s nestrukturovanými a polostrukturovanými sadami distribuovaných dat. Jednou z výhod je, že vývojáři mohou provádět změny databáze za běhu, aniž by to ovlivnilo aplikace, které databázi používají.

3.3.4 Databáze Klíč-Hodnota

Databáze klíč-hodnota poskytuje nejjednodušší možný NoSQL datový model. Data jsou uložena jako pár klíč - hodnota ve slovníku / mapě, kdy klíč je indexem. Hodnota může být například celé číslo, řetězec, struktura JSON

nebo pole. Typické příklady jsou: Redis, Riak, LevelDB. Ukázku databáze klíč-hodnota lze vidět na obrázku č. 3.6.



Obrázek 3.6: Ukázka databáze klíč-hodnota.

Datový model

todo

Výhody

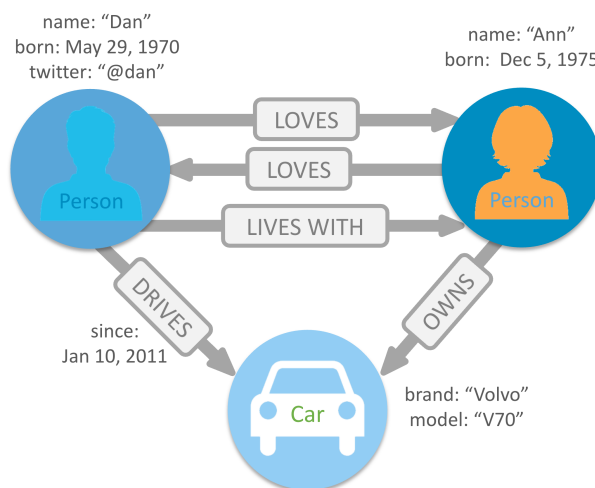
todo

Nevýhody

todo

3.3.5 Grafová databáze

Grafová databáze je typem NoSQL databáze, která je založená na teorii grafů. Data jsou reprezentována jako uzly, hrany zase reprezentují vztahy mezi daty. Graf lze procházet podél určitých typů hran nebo přes celý graf. Procházení spojení nebo relací je velmi rychlé, protože vztahy mezi uzly se nepočítají v době dotazu, ale jsou v databázi trvalé. Typické příklady jsou: Neo4j, OrientDB, Microsoft Azure CosmosDB. Ukázku grafové databáze lze vidět na obrázku č. 3.7.



Obrázek 3.7: Ukázka grafové databáze

Datový model

todo

Výhody

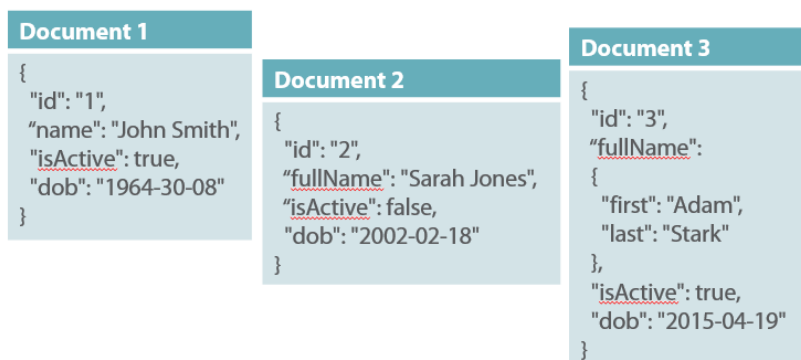
todo

Nevýhody

todo

3.3.6 Dokumentová databáze

Databáze dokumentů jsou typem NoSQL databáze a jsou navrženy pro ukládání, načítání a správu informací orientovaných na dokumenty. Dokumenty jsou obvykle uloženy ve formátu XML, JSON, BSON. Typické příklady jsou: MongoDB, Amazon DocumentDB, Elasticsearch. Ukázku dokumentové databáze lze vidět na obrázku č. 3.8.



Obrázek 3.8: Ukázka dokumentově orientované databáze

Datový model

todo

Výhody

todo

Nevýhody

todo

3.4 Existující řešení

Pro vybrané typy databáze existují mnoho databázových řešení, které lze zmínit. V této kapitole se budeme zabývat především těmi nejznámějšími pro daný typ databáze, a které jsou zdarma ke stažení a používání. Pro každý typ databáze byly vybrány maximálně dvě řešení.

3.4.1 MySQL

MySQL je multiplatformní databáze uplatňující relační databázový model. Komunikace s databází (získávání dat, vytváření objektů, ...) probíhá pomocí jazyka SQL, který je rozšířen o nové funkce. Nejnovější verze MySQL je open-source, což znamená, že kdokoliv může používat a libovolně upravovat MySQL systém, aniž by musel cokoli platit. V případě změny zdrojových kódů je potřeba nastudovat podmínky užívání definované licencí GPL [6].

Od samých počátků bylo MySQL optimalizováno především na rychlost i za cenu některých zjednodušení (například způsob zálohování dat). Díky tomuto lze provozovat jednoduché servery na počítači společně s jinými aplikacemi, případně jiné databáze. Server lze nakonfigurovat tím způsobem, že může využívat veškerou paměť, procesorový čas i vstupně výstupní kapacity.

MySQL server může být využit dvěma způsoby:

- **Klient / server** - vícevláknový SQL server, který podporuje různé back-endy, několik různých klientských programů a knihoven a mnoho dalšího.
- **Věstavěná knihovna** - vícevláknová věstavěná knihovna, kterou lze propojit do své aplikace a získat tím menší, rychlejší a snadněji spravovatelný samostatný produkt.

3.4.2 PostgreSQL

Relační

todo

3.4.3 ConceptBase

OO

todo

3.4.4 LevelDB

KeyValue

todo

3.4.5 Redis

KeyValue

todo

3.4.6 MongoDB

Document

todo

3.4.7 Elasticsearch

Document

todo

3.4.8 Neo4j

Graf

todo

3.5 Jazyky

Databázové jazyky, jinak známé jako dotazovací jazyky, jsou klasifikací programovacích jazyků, které se používají k definování a přístupu k databázím. Pomocí těchto jazyků dokáže uživatel získávat nebo spravovat data v databázích. V dnešní době se jazyky (například SQL) mohou skládat ze čtyř podjazyků, kdy každý slouží k jinému účelu v rámci vykonávání příkazů [3, 10]:

- **Data definition language** (DDL) - DDL umí vytvářet jednotlivé komponenty databázového schématu (tabulky, soubory, indexy, ...), které tvoří strukturu reprezentující organizaci dat v databázi. Dostupné příkazy pro jazyk DDL:
 - **CREATE** - Vytvoření nového objektu (tabulka, index, ...).
 - **ALTER** - Změna struktury objektu.
 - **DROP** - Smazání objektu.

- **RENAME** - Změna názvu objektu.
- **TRUNCATE** - Smazání podobjektů v objektu (například záznamy v tabulce).
- **Data manipulation language (DML)** - DML slouží pro manipulaci s daty, které se nachází v již existující databázi. Dostupné příkazy pro jazyk DML:
 - **SELECT** - Získání záznamů (dat) z tabulky.
 - **INSERT** - Vložení nového záznamu (dat) do tabulky.
 - **UPDATE** - Úprava existujícího záznamu v tabulce.
 - **DELETE** - Smazání záznamu z tabulky.
- **Data control language (DCL)** - Pomocí DCL lze kontrolovat přístupy a práva k datům, které jsou uloženy v databázi. Uživatelé lze nastavit práva k jednotlivým DML příkazům nad tabulkami / procedurami (například uživatel bude mít přístup pouze k příkazu **SELECT** nad tabulkou "TABULKA"). Dostupné příkazy pro jazyk DCL:
 - **GRANT** - Přidání práv uživateli nad danou tabulkou / procedurou.
 - **REVOKE** - Odebrání práv uživateli nad danou tabulkou / procedurou.
- **Transaction control language (TCL)** - TCL spravuje transakce v databázi. Transakce obsahuje jeden či více DML příkazů nad tabulkami, které se vykonávají po sobě. Všechny příkazy musí být úspěšně provedeny, aby bylo možné transakci označit za úspěšnou. Ukázka jedné transakce viz obrázek č. 3.9. Dostupné příkazy pro jazyk TCL:
 - **COMMIT** - Potvrzení transakce, změny provedené v transakci jsou permanentní a nejdou vzít zpět.
 - **ROLLBACK** - Vezme zpět veškerou práci v aktuální transakci. Lze se vrátit na začátek transakce nebo k **SAVEPOINTu**.
 - **SAVEPOINT** - Nastavení bodu v transakci, ke kterému se lze v budoucnu vrátit pomocí **ROLLBACK**.

```
SQL> SAVEPOINT SP1;
Savepoint created.
SQL> DELETE FROM CUSTOMERS WHERE ID=1;
1 row deleted.
SQL> SAVEPOINT SP2;
Savepoint created.
SQL> DELETE FROM CUSTOMERS WHERE ID=2;
1 row deleted.
SQL> SAVEPOINT SP3;
Savepoint created.
SQL> DELETE FROM CUSTOMERS WHERE ID=3;
1 row deleted.
SQL> ROLLBACK TO SP2;
Rollback complete.
```

Obrázek 3.9: Ukázka jedné transakce (bez commitu)

Níže v kapitolách jsou popsány příklady dnešních jazyků.

3.5.1 Structured Query Language

Structured Query Language (SQL) je jazyk pro komunikaci s databázema, v dnešní době standard pro relační databázové systémy. Pomocí SQL příkazů lze například vytvářet nové objekty v databázi, upravovat existující data v tabulkách nebo vytvářet různá integritní omezení a triggerů. Většina existujících databázových systémů používá upravený SQL jazyk, který navíc obsahuje dodatečná rozšíření pro splnění požadavků v jejich systémech.

Syntax

Syntaxe SQL se skládá z unikátního seznamu pravidel a směrnic. Při psaní příkazů zde nehraje roli citlivost písma (příkazy `select` a `SELECT` jsou záměnné). Dotazy lze psát na jednu nebo více řádek, které musí / mohou být zakončené středníkem (záleží na pravidlech používaného systému). Na obrázku č. 3.10 lze vidět příklad dotazu, který získá jména a příjmení uživatelů z tabulky `'user'` s datem narození po roce 2000.

```
SELECT firstname, lastname FROM user WHERE birthdate_year > 2000;
```

Obrázek 3.10: Příklad SQL dotazu.

Dotazy lze zanořovat do sebe, kdy výsledek jednoho dotazu jde použít jako podmínka pro druhý dotaz, viz obrázek č. 3.11.

```
SELECT * FROM user WHERE user.id = (SELECT id_user FROM meal_orders WHERE id = 1);
```

Obrázek 3.11: Příklad zanořeného SQL dotazu.

3.5.2 MongoDB Query Language

MongoDB Query Language (MQL) je jazyk pro získávání dat z MongoDB dokumentových databází. Dotazy zde poskytují jednoduchost v procesu načítání dat z databáze, stejně jako tomu je u SQL. Při provádění dotazů lze také použít kritéria nebo podmínky, kterými lze načíst konkrétní data z databáze. Jazyk také podporuje CRUD operace. Výsledky můžeme třídit, seskupovat, filtrovat a spočítat jejich četnost za pomoci agregační pipeline (zřetěženého zpracování). MQL podporuje transakce více dokumentů [5].

Syntax

Syntaxe MQL je intuitivní a jednoduchá na používání i pro velice složité dotazy, protože ta samá syntaxe se používá i pro uložení dokumentů v databázi. Příklad syntaxe pro vytváření, čtení, úpravu a mazání dokumentů (CRUD) lze vidět na obrázku č. 3.12.

(a) Create

```
db.users.insertOne(  ← collection
{
  name: "sue",        ← field: value
  age: 26,            ← field: value
  status: "pending"   ← field: value } document
})
```

(b) Read

```
db.users.find(
  { age: { $gt: 18 } }, ← collection
  { name: 1, address: 1 } ← query criteria
).limit(5)              ← projection
                        ← cursor modifier
```

(c) Update

```
db.users.updateMany( ← collection
  { age: { $lt: 18 } }, ← update filter
  { $set: { status: "reject" } } ← update action
)
```

(d) Delete

```
db.users.deleteMany( ← collection
  { status: "reject" } ← delete filter
)
```

Obrázek 3.12: Příklady CRUD operací v MongoDB.

3.5.3 Cypher Query Language

Cypher je dotazovací jazyk pro grafovou databázi Neo4j a umožňuje získávat data z grafů. Tento jazyk byl inspirován hlavně SQL - uživatel se zaměřuje pouze na to, jaká data chce získat, ne jak je má získat. Cypher je unikátní v tom, že poskytuje vizuální způsob, jak sladit vzory a vztahy [2].

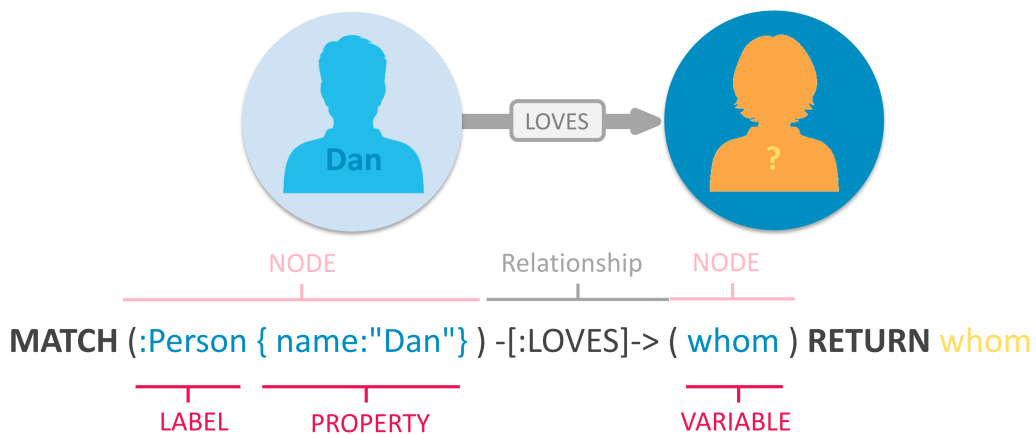
Syntax

Cypher využívá ASCII-art typ syntaxe, což je umění, které pracuje s počítačovým textem jako s výtvarným médiem (například obrázky se skládají ze znaků kódu ASCII). Syntaxi lze vidět na obrázku č. 3.13. Pro jednotlivé uzly se používají kruhové závorky, pro vztah se používá šipka s hranatými závorkama obsahující vztah prvního uzlu s druhým uzlem.

```
(nodes) - [ :ARE_CONNECTED_TO ] -> (otherNodes)
```

Obrázek 3.13: Syntaxe jazyka Cypher.

Na obrázku č. 3.14 lze vidět jednoduchý dotaz, který hledá výsledný uzel pro vstupní uzel, kterým je člověk se jménem 'Dan', a vztahu 'LOVES' mezi uzly.



Obrázek 3.14: Dotaz v jazyce Cypher.

4 Návrh úložiště

Zdroje (které ano, které ne + proč) + udělat stejnou tabulku jak v excelu + zmínit stránku ze které jsem čerpal informace (wipo.int) todo
Nezapomenout zmínit meze let, ve kterých se jednotlivé patenty z daných zemí nachází todo

4.1 Výběr patentů

Při výběru patentů byly stanoveny tři podmínky, které museli být splněny:

- Dostupnost - patenty musí být dostupné z online stránek / databází bez poplatků.
- Datum - patentová přihláška nebo publikace patentu musí být podána alespoň v roce 2000, všechny ostatní patenty budou vyfiltrovány.
- Atributy - všechny patenty musí obsahovat povinné atributy (viz kapitola č. 4.1.2).

4.1.1 Zdroje dat

V dnešním světě existuje několik desítek až stovek patentových zdrojů dat, od webových vyhledávačů v databázi až po plný export databáze s patenty. Velké organizace, jako například EPO, WIPO, USPTO, udržují jedny z největších patentových databází (desítky až stovky milionů patentů), ve kterých lze vyhledávat velké množství informací zdarma za použití webových vyhledávačů na dané stránce organizace. Lze zde najít všechny typy patentů (přihlášky, publikace), národní patenty i patenty registrované například u EPO. V případě exportu databází, USPTO poskytuje plný export svých databází veřejnosti pro libovolné používání, zcela zdarma. Využití těchto zdrojů dat by bylo určitě skvělé, ale tyto zdroje byly nedávno použity a rozebrány v jiné diplomové práci, proto je vhodné se spíše zaměřit na národní zdroje dat patentů.

Národní databáze patentů dané země obsahuje všechny národní patenty, některé dokonce i patenty z jiných zemí registrovaných u EPO.

Země	Patentový úřad	Zkratka
Anglie	Intellectual Property Office	IPO
Arménie	Intellectual Property Office	-
Austrálie	IP Australia	-
Bělorusko	National Center of Intellectual Property	NCIP
Bulharsko	Patent Office of Republic of Bulgaria	-
Česko	Industrial Property Office of the Czech Republic	-
Čína	China National Intellectual Property Administration	CNIPA
Dánsko	Danish Patent and Trademark Office	-
Egypt	Egyptian Patent Office	-
Estonsko	The Estonian Patent Office	-
Filipíny	Intellectual Property Office of the Philippines	IPOPHL
Finsko	Finnish Patent and Registration Office	PRH
Francie	National Institute of Industrial Property	INPI
Hong Kong	Intellectual Property Department	-
Chorvatsko	State Intellectual Property Office of the Republic of Croatia	SIPO
Indie	Office of the Controller General of Patents, Designs and Trade Marks	-
Indonésie	Directorate General of Intellectual Property	DGIP
Irsko	Intellectual Property Office of Ireland	IPOI
Island	Icelandic Intellectual Property Office	ISIPO
Israel	The Israel Patent Office	ILPO
Itálie	Directorate General for the Protection of Industrial Property	-
Japonsko	Japan Patent Office	JPO
Jižní Korea	Korean Intellectual Property Office	KIPO
Kanada	Canadian Intellectual Property Office	CIPO

Tabulka 4.1: Národní patentové úřady a jejich zkratky, část první

Země	Patentový úřad	Zkratka
Kuba	Cuban Industrial Property Office	OCPI
Litva	State Patent Bureau of the Republic of Lithuania	-
Lotyšsko	Patent Office of the Republic of Latvia	-
Maďarsko	Hungarian Intellectual Property Office	HIPO
Malajsie	Intellectual Property Corporation of Malaysia	MyIPO
Mexiko	Instituto Mexicano De La Propiedad Industrial	IMPI
Moldova	State Agency on Intellectual Property	AGEPI
Německo	German Patent and Trade Mark Office	DPMA
Nizozemsko	Netherlands Patent Office	-
Norsko	Norwegian Industrial Property Office	NIPO
Nový Zéland	Intellectual Property Office of New Zealand	IPONZ
Peru	National Institute for the Defense of Competition and Protection of Intellectual Property	INDECOPI
Polsko	Urząd Patentowy Rzeczypospolitej Polskiej	UPRP
Portugalsko	Portuguese Institute of Industrial Property	-
Rakousko	Austrian Patent Office	-
Rumunsko	State Office for Inventions and Trademarks	OSIM
Rusko	Federal Service for Intellectual Property	Rospatent
Řecko	Hellenic Industrial Property Organization	HIPO
Singapur	Intellectual Property Office of Singapore	IPOS
Slovensko	Industrial Property Office of the Slovak Republic	-
Slovinsko	Slovenian Intellectual Property Office	SIPO
Srbsko	Intellectual Property Office of the Republic of Serbia	-
Španělsko	Spanish Patent and Trademark Office	OEPM
Švédsko	Swedish Intellectual Property Office	PRV
Švýcarsko	Swiss Federal Institute of Intellectual Property	-
Turecko	Turkish Patent and Trademark Office	Turkpatent
Ukrajina	Ukrainian Intellectual Property Institute	Ukrpatent

Tabulka 4.2: Národní patentové úřady a jejich zkratky, část druhá

4.1.2 Atributy

Povinné atributy

Země	Název patentu	Rok přihlášky / publikace	Autor	ID patentu
Kanada	x	x	x	x
Česko	x	x	-	x
Litva	x	x	x	x
Portugalsko	x	x	x	x
Španělsko	x	x	x	x
Švédsko	-	x	-	x
Izrael	x	x	x	x
Itálie	x	x	x	x
Mexiko	x	x	x	x
Polsko	x	x	-	-
Anglie	x	x	x	x
Rusko	x	x	x	x
Peru	x	x	x	x
Francie	x	x	x	x

Tabulka 4.3: Povinné atributy nacházející se v dostupných patentech

Nepovinné atributy

Země	Abstrakt	Slovník	Reference	Žadatel
Kanada	-	-	-	x
Česko	x	-	x	-
Litva	x	-	-	x
Portugalsko	x	-	-	x
Španělsko	x	-	-	x
Švédsko	x	-	-	-
Izrael	-	-	-	x
Itálie	-	-	-	x
Mexiko	x	-	-	x
Polsko	x	x	-	-
Anglie	-	-	-	-
Rusko	-	-	-	-
Peru	-	-	-	-
Francie	x	-	x	x

Tabulka 4.4: Nepovinné atributy nacházející se v dostupných patentech, část první

Země	Adresa	Rodina patentů	Obor	Fulltext
Kanada	x	-	x	-
Česko	-	-	x	-
Litva	-	-	x	-
Portugalsko	-	-	x	-
Španělsko	x	-	x	x
Švédsko	-	-	x	x
Izrael	x	-	-	-
Itálie	-	-	-	-
Mexiko	-	-	x	-
Polsko	-	-	-	-
Anglie	-	-	x	-
Rusko	-	-	-	-
Peru	-	-	x	-
Francie	-	-	x	-

Tabulka 4.5: Nepovinné atributy nacházející se v dostupných patentech, část druhá

4.1.3 Závěr průzkumu

Rovnou udělat kapitolu, ve který se aplikují všechny podmínky + se sepíše souhrn počtu patentů, z jakých zemí atp.

4.2 Výběr databáze

todo

4.2.1 Výběr typu databáze

4.2.2 Výběr z existujících řešení

4.2.3 Závěr průzkumu

5 Implementace úložiště

todo

5.1 Adresářová struktura

todo

5.1.1 Patenty

5.1.2 Docker

5.2 Implementace databáze

todo

5.2.1 MySQL

5.2.2 MongoDB

5.3 Výsledné úložiště

todo

5.3.1 Technologické požadavky

5.3.2 Docker

Images, scripty

Import dat (skripty, data soubory, ...)

5.3.3 Inicializace MySQL

5.3.4 Inicializace MongoDB

6 Rozšiřitelnost úložiště

Zadání diplomové práce sice splněno bylo, ale v blízké budoucnosti mohou být požadavky na modul změněny. Jako příklad lze uvést podporu přidávání nových patentů do databází, zjištění autorů pro české patenty, automatické stahování dat z již ověřených patentových zdrojů. V této kapitole jsou popsány 3 možné návrhy na rozšíření modulu ohledně importu dat do již existujících databází.

6.1 Přidávání nových patentů

Cílem tohoto rozšíření by bylo automatické přidávání patentů z datových souborů jak do MySQL databáze, tak i do Mongo.

Rozšíření by se dalo realizovat jako aplikace ve vyšším programovacím jazyku (např. Java, C), kdy vstupem do aplikace by byl soubor v datovém formátu JSON/XML/CSV a jiné. Vstupní soubor by se následně:

- převedl na JSON řetězec (v případě že soubor není ve formátu JSON) a vložil do Mongo databáze
- rozparsoval a extrahovali by se všechny atributy, které se ukládají v MySQL databázi (viz mysql kapitola)

TODO

Jelikož je dost časté, že každý národní zdroj dat používá odlišnou strukturu patentu, tak bude potřeba aplikaci neustále upravovat (ať už v rámci přidávání nových zdrojů, nebo v případě změny struktury patentu u již podporovaných zdrojů).

Jako další velký problém lze zmínit extrakci atributů patentu ze souborů. Tím, že různé patentové soubory mají odlišnou strukturu, to znamená hloubku zanoření specifických elementů, jiné názvy elementů, tak bude obtížné naimplementovat řešení extrakce pro všechny soubory. Tento problém by se dal řešit tak, že se vytvoří soubory se slovníkama, které by obsahovaly názvy elementů pro daný atribut. Slovníky by se následně použily při extrakci.

6.2 Zjišťování autorů pro české patenty

Český národní patentový úřad poskytuje data o českých patentech, které ale neobsahují autora ani instituci. Pro zjištění autora nebo instituce, která

patent registrovala, je nutné použít oficiální vyhledávač. Cílem tohoto rozšíření by bylo vytvořit aplikaci ve vyšším programovacím jazyku, která se pro všechny české patenty bude snažit najít jejich autory za pomoci využití prohledávačů webů (web crawler). Postupů řešení může být mnoho:

- Zjišťování autorů by se provedlo pro všechny existující české patenty v databázi. Z MySQL databáze se zjistí všechny identifikátory pro české patenty, které se následně použijí jako vstup pro web crawler.
- Zjišťování autorů by se provedlo pro patent/y uložené v souboru, kdy aplikace by pro všechny patenty v souboru zjistila autory a následně je dopsala do příslušného elementu patentu v daném souboru.
- Stejný postup jako předchozí s tím rozdílem, že po zjištění autora se patent rovnou přidá do MySQL i Mongo databáze.

6.3 Automatické stahování dat z ověřených zdrojů

Cílem tohoto rozšíření by bylo automatické stahování dat (případně i jejich parsování) z ověřených zdrojů. Ověřené zdroje by byly uloženy například v XML souboru, kdy každý zdroj by měl tyto položky:

- **Název země**
- **URL** - URL zdroje dat, na které lze stáhnout data.
- **XPath** - XPath výraz, pomocí kterého lze ze stránky vyfiltrovat a získat odkazy ke stažení dat
(např. `/html/body//a[contains(@href,'example')]/@href`)
- **Poslední verze** - Název / číslo poslední stažené verze.

XML soubor by byl následně zpracován pomocí aplikace (např. Java, C#), která by následně pro každý zdroj dat provedla následující kroky:

1. Získání seznamu odkazů na zdroje dat.
2. Stažení všech zdrojů dat, jejichž verze je větší než aktuálně uložená verze v XML.
3. V tomto bodě se dá naimplementovat cokoliv - např. lze uložená data extrahovat ze ZIP souborů, importovat patenty do databází (viz kapitola č. 6.1), pouze notifikace o stažení několika nových souborů z daty a mnoho dalšího.

4. Aktualizace verze v XML souboru.

Automatizace stahování dat by spočívala ve spouštění aplikace pro stahování dat v pravidelných intervalech (např. každé druhé úterý v 17:00). Jako příklad lze uvést použití pipeline na Jenkins serveru, který bude spouštět z lokálního uložště spustitelnou aplikaci v daný čas (pomocí CRON). Po vykonání celého procesu může Jenkins poslat email o stavu posledního spuštění (zda se spuštění povedlo, kolik souborů byl schopen stáhnout pro jaké země, ...). Samozřejmě bohatě postačí i použití plánovače v operačním systému.

7 Ověření efektivního vytěžování

K ověření efektivního vytěžování bylo připraveno několik scénářů jak pro SQL, tak i pro Mongo + Elasticsearch.

Napsat referenční stroj na kterém se testovalo - CPU, RAM, ...

todo

7.1 Mongo + Elasticsearch

7.2 MySQL

Pro MySQL bylo připraveno 10 scénářů, které testují všechny vytvořené tabulky v databázi. Každý scénář obsahuje textový popis, SQL příkaz, rychlost vykonání příkazu a ukázkou výsledků.

vypsat počet hodnot v tabulkách

7.2.1 Scénář č.1

Textový popis: Deset nejčastěji patentujících institucí v Izraeli v roce 2015.

SQL:

```
select count(*), count(*) * 100.0 / ((select count(*) from inventors) * 1.0) as percentage, inventors.inventor from inventors left outer join patents on inventors.id_patent = patents.id where YEAR(patents.patent_date) = 2015 and patents.patent_id like '%IL%' group by inventors.inventor order by count(*) desc, percentage desc LIMIT 10;
```

Rychlost vykonání dotazu:

TODO

Výsledek dotazu:

TODO

count(*)	percentage	inventor
39	0.00119	DOW AGROSCIENCES LLC
29	0.00088	F. HOFFMANN-LA ROCHE AG
29	0.00088	GENENTECH, INC.
29	0.00088	NOVARTIS AG
24	0.00073	RAYTHEON COMPANY
22	0.00067	QUALCOMM INCORPORATED
20	0.00061	BIOSENSE WEBSTER (ISRAEL) LTD.
17	0.00052	MICROSOFT CORPORATION
17	0.00052	SANOFI
16	0.00049	YERKES, CARLA N.

Obrázek 7.1: Ukázka výsledku dotazu pro scénář č.1

7.2.2 Scénář č.2

Textový popis: Tři nejméně patentované obory v Kanadě od roku 2010.

SQL:

```
select count(*), count(*) * 100.0 / ((select count(*) from classification
) * 1.0) as percentage, classification.section from classification
left outer join patents on patents.id = classification.id_patent
where YEAR(patents.patent_date) >= 2010 and patents.patent_id like '%
CA%' group by classification.section order by count(*) asc,
percentage asc LIMIT 5;
```

Rychlost vykonání dotazu: _____

TODO

Výsledek dotazu: _____

TODO

count(*)	inventor
8	Gould Nigel
4	Philips Andrew
4	Lee Sangik
3	Weisman Paul Thomas
3	Dreher Andreas Josef
3	Sivik Mark Robert
3	Park Bio
3	Kim Jeongyun
3	Hamad-Ebrahimpour Alyssandrea Hope
3	Kim Seonghwan
3	Gordon Gregory Charles
3	Trokhan Paul Dennis
2	Watson David John
2	Waite David
2	Payne John Lynn

Obrázek 7.2: Ukázka výsledku dotazu pro scénář č.2

7.2.3 Scénář č.3

Textový popis: Dvacet nejčastějších klasifikací patentů za rok 2008 ve Španělsku.

SQL:

```
select count(*), count(*) * 100.0 / ((select count(*) from classification
) * 1.0) as percentage, classification.section, classification.class,
classification.subclass from classification left outer join patents
on patents.id = classification.id_patent where YEAR(patents.
patent_date) = 2008 and patents.patent_id LIKE '%ES%' group by
classification.section, classification.class, classification.subclass
order by count(*) desc, percentage desc LIMIT 20;
```

Rychlost vykonání dotazu: _____

TODO

Výsledek dotazu: _____

TODO

count(*)	percentage	section	class	subclass
73	0.00481	B	65	D
57	0.00376	E	04	G
47	0.00310	A	61	K
45	0.00296	E	04	B
43	0.00283	G	01	N
34	0.00224	E	06	B
33	0.00217	F	24	J
30	0.00198	A	23	L
27	0.00178	C	02	F
27	0.00178	A	47	C
27	0.00178	B	60	R
27	0.00178	E	04	H
26	0.00171	A	47	L
26	0.00171	D	06	F
25	0.00165	F	03	D
24	0.00158	A	01	K
23	0.00152	H	05	B
23	0.00152	B	01	D
22	0.00145	B	65	B
22	0.00145	E	04	C

Obrázek 7.3: Ukázka výsledku dotazu pro scénář č.3

7.2.4 Scénář č.4

Textový popis: Deset autorů s největším počtem patentů ze všech zemí.

SQL: _____

divnej do-
taz

```
select count(*), count(*) * 100.0 / ((select count(*) from inventors) *
1.0) as percentage, inventors.inventor from inventors left outer join
```

```
patents on patents.id = inventors.id_patent group by inventors.
inventor order by count(*) desc, percentage desc LIMIT 10;
```

Rychlost vykonání dotazu: _____

TODO

Výsledek dotazu: _____

TODO

count(*)	inventor
8	Gould Nigel
4	Philips Andrew
4	Lee Sangik
3	Weisman Paul Thomas
3	Dreher Andreas Josef
3	Sivik Mark Robert
3	Park Bio
3	Kim Jeongyun
3	Hamad-Ebrahimpour Alyssandrea Hope
3	Kim Seonghwan
3	Gordon Gregory Charles
3	Trokhan Paul Dennis
2	Watson David John
2	Waite David
2	Payne John Lynn

Obrázek 7.4: Ukázka výsledku dotazu pro scénář č.4

7.2.5 Scénář č.5

Textový popis: Pět nejméně používaných jazyků pro patenty za rok 2003.

SQL:

```
select count(*), count(*) * 100.0 / ((select count(*) from patents) *
1.0) as percentage, patents.language from patents where patents.
language not like '%-%' group by patents.language order by count(*)
asc, percentage asc LIMIT 5;
```

Rychlost vykonání dotazu: _____

TODO

Výsledek dotazu: _____

TODO

count(*)	inventor
8	Gould Nigel
4	Philips Andrew
4	Lee Sangik
3	Weisman Paul Thomas
3	Dreher Andreas Josef
3	Sivik Mark Robert
3	Park Bio
3	Kim Jeongyun
3	Hamad-Ebrahimipour Alyssandrea Hope
3	Kim Seonghwan
3	Gordon Gregory Charles
3	Trokhan Paul Dennis
2	Watson David John
2	Waite David
2	Payne John Lynn

Obrázek 7.5: Ukázka výsledku dotazu pro scénář č.5

7.2.6 Scénář č.6

Textový popis: Deset Institucí / autorů s patenty pokrývající největší množství oborů.

SQL:

```
select count(distinct classification.section), count(*) * 100.0 / ((
    select count(*) from inventors) * 1.0) as percentage, inventors.
inventor from inventors left outer join classification on
classification.id_patent = inventors.id_patent where section is not
null group by inventors.inventor order by count(distinct
classification.section) desc, percentage desc LIMIT 10;
```

Rychlost vykonání dotazu: _____ TODO

Výsledek dotazu: _____ TODO

count(*)	inventor
8	Gould Nigel
4	Philips Andrew
4	Lee Sangik
3	Weisman Paul Thomas
3	Dreher Andreas Josef
3	Sivik Mark Robert
3	Park Bio
3	Kim Jeongyun
3	Hamad-Ebrahimpour Alyssandrea Hope
3	Kim Seonghwan
3	Gordon Gregory Charles
3	Trokhan Paul Dennis
2	Watson David John
2	Waite David
2	Payne John Lynn

Obrázek 7.6: Ukázka výsledku dotazu pro scénář č.6

7.2.7 Scénář č.7

Textový popis: Pět zemí s nejvíce patenty od roku 2018.

SQL:

```
select count(*), count(*) * 100.0 / ((select count(*) from patents) *
1.0) as percentage, patents.country from patents where YEAR(patents.
patent_date) >= 2018 group by patents.country order by count(*) desc,
percentage desc LIMIT 5;
```

Rychlost vykonání dotazu: _____

TODO

Výsledek dotazu: _____

TODO

count(*)	inventor
8	Gould Nigel
4	Philips Andrew
4	Lee Sangik
3	Weisman Paul Thomas
3	Dreher Andreas Josef
3	Sivik Mark Robert
3	Park Bio
3	Kim Jeongyun
3	Hamad-Ebrahimpour Alyssandrea Hope
3	Kim Seonghwan
3	Gordon Gregory Charles
3	Trokhan Paul Dennis
2	Watson David John
2	Waite David
2	Payne John Lynn

Obrázek 7.7: Ukázka výsledku dotazu pro scénář č.7

7.2.8 Scénář č.8

Textový popis: Nejvíce používaný jazyk pro patenty ve Francii.

SQL:

```
select count(*), count(*) * 100.0 / ((select count(*) from patents) * 1.0) as percentage, patents.language from patents where patents.patent_id like '%FR%' group by patents.language order by count(*) desc, percentage desc;
```

Rychlost vykonání dotazu: _____

TODO

Výsledek dotazu: _____

TODO

count(*)	inventor
8	Gould Nigel
4	Philips Andrew
4	Lee Sangik
3	Weisman Paul Thomas
3	Dreher Andreas Josef
3	Sivik Mark Robert
3	Park Bio
3	Kim Jeongyun
3	Hamad-Ebrahimpour Alyssandrea Hope
3	Kim Seonghwan
3	Gordon Gregory Charles
3	Trokhan Paul Dennis
2	Watson David John
2	Waite David
2	Payne John Lynn

Obrázek 7.8: Ukázka výsledku dotazu pro scénář č.8

7.2.9 Scénář č.9

Textový popis: Patnáct nejčastěji patentujících institucí / autorů v Anglii v textilním oboru za rok 2013.

SQL:

```
select count(*), count(*) * 100.0 / ((select count(*) from inventors) * 1.0) as percentage, inventors.inventor from inventors left outer join patents on patents.id = inventors.id_patent left outer join classification on classification.id_patent = patents.id where classification.section like '%D%' and patents.patent_id like '%GB%' and YEAR(patents.patent_date) = 2013 group by inventors.inventor order by count(*) desc, percentage desc LIMIT 15;
```

Rychlost vykonání dotazu: _____

TODO

Výsledek dotazu:

TODO

count(*)	inventor
8	Gould Nigel
4	Philips Andrew
4	Lee Sangik
3	Weisman Paul Thomas
3	Dreher Andreas Josef
3	Sivik Mark Robert
3	Park Bio
3	Kim Jeongyun
3	Hamad-Ebrahimpour Alyssandrea Hope
3	Kim Seonghwan
3	Gordon Gregory Charles
3	Trokhan Paul Dennis
2	Watson David John
2	Waite David
2	Payne John Lynn

Obrázek 7.9: Ukázka výsledku dotazu pro scénář č.9

8 Závěr

todo

Zkratky

ACID Atomicity, Consistency, Isolation, Durability 4, 7

ASCII American Standard Code for Information Interchange 16

JSON Binary JSON 9

CRUD Create, Read, Update, Delete 14, 15

CSV Comma-separated values 24

DBMS Database Management Systems 3

EPO European Patent Office 17

GPL GNU General Public License 10

JSON JavaScript Object Notation 7, 9, 24

SQL MongoDB Query Language 14

SQL Structured Query Language 7, 10, 11, 13–15

URL Uniform Resource Locator 25

USPTO United States Patent and Trademark Office 17

WIPO World Intellectual Property Organization 17

XML Extensible Markup Language 9, 24–26

Literatura

- [1] *What Is a Database?* [online]. Oracle. [cit. 21.04.2022]. Dostupné z: <https://www.oracle.com/database/what-is-database/>.
- [2] *Cypher Query Language* [online]. Neo4j. [cit. 26.04.2022]. Dostupné z: <https://neo4j.com/developer/cypher/>.
- [3] *Types of Database Languages and Their Uses (Plus Examples)* [online]. indeed, 2021. [cit. 21.04.2022]. Dostupné z: <https://www.indeed.com/career-advice/career-development/database-languages>.
- [4] *The Types of Databases (with Examples)* [online]. Matillion, 2018. [cit. 22.04.2022]. Dostupné z: <https://www.matillion.com/resources/blog/the-types-of-databases-with-examples>.
- [5] *What is a MongoDB Query?* [online]. MongoDB, 2021. [cit. 26.04.2022]. Dostupné z: <https://www.geeksforgeeks.org/what-is-a-mongodb-query/>.
- [6] *What is MySQL?* [online]. MySQL. [cit. 26.04.2022]. Dostupné z: <https://dev.mysql.com/doc/refman/8.0/en/what-is-mysql.html>.
- [7] GULATI, V. *Relational Model in DBMS* [online]. Scaler, 2022. [cit. 26.04.2022]. Dostupné z: <https://www.scaler.com/topics/dbms/relational-model-in-dbms/>.
- [8] LUTKEVICH, B. *database (DB)* [online]. Tech Target, 2021. [cit. 21.04.2022]. Dostupné z: <https://www.techtarget.com/searchdatamanagement/definition/database>.
- [9] PETERSON, R. *What is a Database? Definition, Meaning, Types with Example* [online]. Guru99, 2022. [cit. 21.04.2022]. Dostupné z: <https://www.guru99.com/introduction-to-database-sql.html>.
- [10] SINGH, C. *DBMS languages* [online]. BeginnersBook, 2015. [cit. 21.04.2022]. Dostupné z: <https://beginnersbook.com/2015/04/dbms-languages/>.

A Uživatelská dokumentace

B Vzhled modulů