Katedra informatiky
Přírodovědecká fakulta
Univerzita Palackého v Olomouci

BAKALÁŘSKÁ PRÁCE

Vizuální tvorba databázových dotazů

2020

Petr Unzeitig

Vedoucí práce: Mgr. Petr Krajča, Ph.D.

Studijní obor: Aplikovaná informatika, prezenční forma

Bibliografické údaje

Autor:	Petr Unzeitig
Název práce:	Vizuální tvorba databázových dotazů
Typ práce:	bakalářská práce
Pracoviště:	Katedra informatiky, Přírodovědecká fakulta, Univerzita Palackého v Olomouci
Rok obhajoby:	2020
Studijní obor:	Aplikovaná informatika, prezenční forma
Vedoucí práce:	Mgr. Petr Krajča, Ph.D.
Počet stran:	43
Přílohy:	1 CD/DVD
Jazyk práce:	český

Bibliograhic info

Author:	Petr Unzeitig
Title:	Visual Query Builder
Thesis type:	bachelor thesis
Department:	Department of Computer Science, Faculty of Science, Palacký University Olomouc
Year of defense:	2020
Study field:	Applied Computer Science, full-time form
Supervisor:	Mgr. Petr Krajča, Ph.D.
Page count:	43
Supplements:	1 CD/DVD
Thesis language:	Czech

Anotace

V rámci této bakalářské práce byla vyvinuta aplikace určena koncovým uživatelům pro snadnou tvorbu dotazů v relačních databázových systémech. V práci je popsán relační model dat a systém pro jeho manipulaci pomocí relační algebry. Další část práce je věnována základům dotazovacího jazyka SQL, jenž tvoří prostředek pro práci s daty v relačních databázích. Dále je popsán vývoj aplikace samotné. Definice požadavků, použitých technologií a postup při tvorbě výsledného produktu. V neposlední řadě je vysvětleno její používaní v uživatelské dokumentaci.

Synopsis

As part of this bachelor thesis was developed an application, which is intended for end users to help them with easy query creation in relational database systems. The work describes a relational data model and a system for its manipulation using relational algebra. The next part of the work is devoted to the basics of SQL query language, which is a tool for working with data in relational databases. Also, the development of the application itself is described. Definition of requirements, used technologies and process of creating the final product. Last but not least, is explained its use in the user documentation.

Klíčová slova: relační model; relační algebra; databáze; SQL; SŘBD

Keywords: relational model; relational algebra; database; SQL; DBMS

Děkuji vedoucímu této bakalářské práce panu Mgr. Petru Krajčovi Ph.D. za odbornou pomoc a cenné rady. Děkuji také mé rodině a blízkým za trpělivost a podporu.

Místopřísežně prohlašuji, že jsem celou práci včetně příloh vypracoval samostatně a za použití pouze zdrojů citovaných v textu práce a uvedených v seznamu literatury.

datum odevzdání práce podpis autora

Obsah

Seznam obrázků

Seznam tabulek

Seznam zdrojových kódů

Úvod

V dnešní době je při vývoji softwaru kladen důraz na bezpečné ukládání větších množství dat. Proto je nezbytné, aby šlo snadným způsobem získat požadovaná data zpět a to ve formě, která je pro člověka co nejčitelnější. Tato úloha je zastupována databázovými systémy. Jsou reprezentovány prostřednictvím různých datových modelů, které se liší v mnoha ohledech. Dnes je nejpopulárnější relační model dat. Jedná se o jednoduchý a přesto výkonný model, který používají organizace všech druhů a velikostí. Relační databáze jsou vhodné pro ukládání veškerých typů informací, ve kterých se datové body vztahují k sobě navzájem a musí být řízeny konzistentním, na pravidlech založeným způsobem.

K tomu, aby se uživatelé dostali pohodlně ke svým datům a mohli s nimi manipulovat je nutné určité softwarové vybavení, které tuto práci umožňuje. Proto je potřeba vybrat a využívat vhodný systém řízení báze dat (SŘBD), který tvoří rozhraní mezi aplikačními programy a databázovými daty.

Různé programy, aplikace a webové servery využívají k uchování dat právě relační databáze. K prezentaci dat koncovým uživatelům využívají strukturovaný dotazovací jazyk SQL. Skládáním a vyhodnocováním databázových dotazů jsou schopni zprostředkovat informace na uživatelském rozhraní, se kterým člověk pracuje. Samotný uživatel se tak pomocí interakce s rozhraním daného softwaru stává součástí vyhodnocovacího procesu, jehož výsledkem jsou data obdržená z databáze.

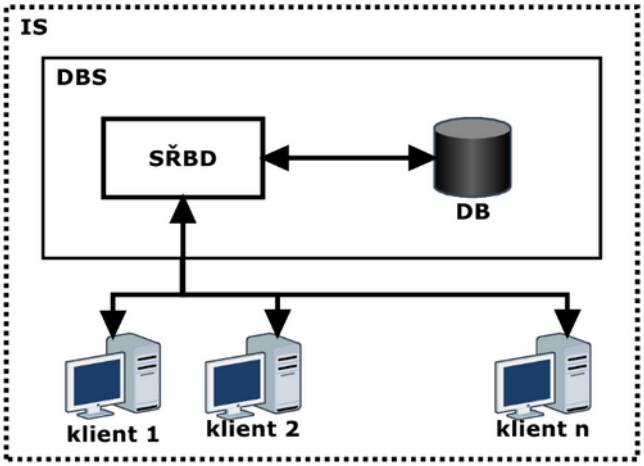
Zaměstnanci ve firmách často potřebují nahlédnout do infrastruktury a pracovat s daty, které se nacházejí v jejich databázi. Musí k tomu využít dotazovací jazyk, jehož obsluha může být složitá.

Z výše zmíněných důvodů je cílem práce vytvořit zaměřený grafický nástroj pro koncové uživatele, který umožní snadnou tvorbu dotazů pro relační databázové systémy. Výsledná aplikace by měla vhodnými vizuálními prostředky umožnit vytvářet jednoduché i složité dotazy zahrnující projekci, restrikci, spojení, množinové operace, agregaci, případně další operace. Navíc by aplikace neměla být závislá na konkrétním SŘBD.

1 Databázový systém

Databázový systém je důležitou stavební složkou pro tvorbu a kvalitní fungování softwaru. Webové stránky, aplikační programy, desktopové a jiné aplikace jsou většinou vyvíjeny právě s databázovým systémem. Tvoří tak základ informačních systémů. Slouží k uchování informací, které lze následně libovolně používat [10].

Databázový systém se skládá ze dvou hlavních částí. Ze samotné databáze, ve které jsou data uložena a ze systému označovaného jako systém řízení báze dat (SŘBD), který tvoří rozhraní mezi daty a uživatelem [8].



Obrázek 1: Struktura databázového systému

Na obrázku 1 jsou zobrazeny jednotlivé části databázového systému, které tvoří jeho strukturu [10]. Jak je vidět, klient (uživatel) může komunikovat s databází pomocí SŘBD oběma směry. Umožňuje tedy data z databáze číst i zapisovat.

1.1 Databáze

Předchůdcem databází byly papírové kartotéky, které stejně jako dnešní databáze uchovávaly jednotlivé záznamy specifikované podle různých vlastností. Lidé měli možnost záznamy v kartotékách uspořádat, třídit, upravovat nebo i vytvořit a vložit nové. Samozřejmě tyto operace musely být prováděny ručně a mnohdy byly velmi zdlouhavé. Proto se dnes klade důraz na rychlé a snadné provádění databázových operací [5].

Databáze, jak je známe dnes, sahají až do 50. let 20. století, kdy vznikl první databázový systém, a to ve formě hierarchického modelu. V tomto modelu jsou jednotlivé záznamy uspořádány ve stromové struktuře, ve vztazích rodič a potomek. Existuje celá řada modelů dat, přes síťový až po objektový [11]. Ale tato práce se zabývá tím nejrozšířenějším a nejpoužívanějším, a to sice modelem relačním, který si blíže popíšeme v kapitole 2.

Pojmem databáze tedy rozumíme množinu perzistentních dat uložených v počítači obsahující záznamy, které jsou organizované pomocí společných rysů.

V databázi mohou být uložena data evidující např.:

- studenty, učitele a jiné pracovníky na univerzitě,
- zboží a jednotlivé objednávky v internetovém obchodě,
- knihy a informace o jejich zapůjčení ve veřejné knihovně,
- filmy, seriály a videa v internetové televizi atd.

1.2 SŘBD

Jak už bylo nastíněno na začátku kapitoly, pojem SŘBD je zkratka pro systém řízení báze dat, který umožňuje uživateli manipulovat s databází a také s daty v ni uloženými. Tento pojem je často zaměňován s pojmem databázový systém. Pro nás je však SŘBD důležitou součástí databázového systému.

Systém řízení báze dat je softwarový celek, zajišťující perzistentní uložení dat a jejich kompletní údržbu. Zajišťuje uživateli, či aplikacím bezpečný přístup k manipulaci s daty vhodným dotazovacím jazykem. Jeho základním účelem je definice, konstrukce samotné databáze, možnost vytvářet, a nebo měnit datové struktury. Dále umožňuje data vkládat, aktualizovat a nebo také jejich vyhledávání, výběr a prezentování [10].

Dnes už existuje nespočet SŘBD, které se liší v mnoha ohledech, ale jejich jádro zůstává stejné. Webová stránka DB-Engines uvádí měsíčně aktualizované údaje o popularitě mnoha SŘBD. Následující seznam obsahuje některé vybrané SŘBD z tohoto severu, seřazené dle jejich uživatelské popularity k dubnu 2020.

1. Oracle
2. MySQL
3. Microsoft SQL Server
4. PostgreSQL
5. MongoDB
6. SQLite
7. MariaDB
8. Firebird[3]

Mimo práci s daty a datovými strukturami je SŘBD zodpovědné za fyzické uložení dat ve vnější části počítače. Pro tyto účely, musí být každý systém spuštěn na odpovídajícím serveru, na kterém závisí výkon výsledné aplikace.[7]

2 Relační model

Relační model dat je kolekce metod a technik pro správu databází, které pracují s určitým množstvím dat. Tyto data se pro jednoduchost reprezentují formou tabulek, kde jednotlivé řádky odpovídají entitám (student, produkt) a sloupce určují jejich vlastnosti (atributy). Například pokud potřebujeme ukládat informace o skupině studentů na univerzitě, mohou být data spravována ve struktuře, která je zobrazena v tabulce 1.

Tabulka 1: Student - tabulka studentů

jméno	**příjmení**	**narození**	**obor**	**ročník**	**status studenta**
Petr	Novák	4.2.1998	Informatika	1	pravda
Lucie	Nová	18.6.1994	Chemie	3	nepravda
Josef	Němec	3.11.1997	Ekonomika	2	pravda
Jaromír	Němec	8.4.1998	Geografie	2	pravda
Lukáš	Svoboda	18.6.1995	Informatika	2	pravda
Marie	Novotná	3.12.1998	Informatika	1	pravda

Relace se v relačním modelu od tabulek liší. Přesněji řečeno, tabulka reprezentuje relaci nad relačním schématem jen tehdy, pokud je v první normální formě (1NF).

Tabulka je v 1NF, pokud splňuje následujících pět podmínek:

1. neuvažuje žádné uspořádání řádků,
2. neuvažuje žádné uspořádání sloupců,
3. neobsahuje duplicitní řádky,
4. na každé pozici dané průsečíkem sloupce a řádku se nachází právě jedna hodnota daného typu,
5. všechny sloupce jsou regulární (řádky nemají žádné skryté hodnoty).

Formálně lze záhlaví tabulky chápat jako množinu dvojic atribut-doména, kde atribut odpovídá názvu sloupce a doména množině přípustných hodnot. Tato množina dvojic se označuje jako relační schéma. Přičemž konečná a spočetná množina všech atributů se označuje Y.

Tělo tabulky odpovídá množině n-tic, kde každá n-tice reprezentuje řádek v tabulce odpovídající relace [2].

S ohledem na zavedenou tabulku *Student* na začátku kapitoly lze uvést například n-tici, která je zobrazena v tabulce 2.

Tabulka 2: n-tice

jméno	**příjmení**	**narození**	**obor**	**ročník**	**status studenta**
Petr	Novák	4.2.1998	Informatika	1	pravda

Pro formalizaci relace nad relačním schématem je nezbytné zavést pojem obecný kartézský součin. Pro systém množin A_i, které jsou indexovány z množiny indexů I je obecný kartézský součin množin $A_i (i \in I)$ množina všech zobrazení $f : I \to \cup_{i \in I} A_i$ takových, že pro každé $i \in I$ platí $f(i) \in A_i$.

Pro relační schéma $R \subseteq Y$, atributy $y \in R$ a jejich domény D_y $(y \in Y)$ je relace D nad relačním schématem R libovolná konečná podmnožina obecného kartézského součinu domén D_y, kde indexy jsou atributy y.

Počet atributů $y \in R$ označovaný jako $|R|$ se nazývá stupeň relace D. Počet n-tic $t \in D$ označovaný jako $|D|$ se nazývá velikost relace D [6].

2.1 SQL

S relačními databázemi je úzce spojen pojem SQL. Je to prostředek pro organizaci, správu a prezentaci získaných dat uložených v databázi. Tato zkratka označuje strukturovaný dotazovací jazyk z anglického *Structured Query Language*. Je integrován v SŘBD a slouží pro komunikaci s databází.[4]

Jeho verze se mohou v jednotlivých SŘBD lišit, ale jistý základ jazyka mají všechny stejný. Lze jej rozdělit na tři části, podle typu uživatele, který ho používá.

Jedna část jazyka je určena administrátorům databáze. Ta zahrnuje dotazy pro vytvoření a modifikaci datových struktur a stojí na samotném vzniku databáze. Jedná se o jazyk pro definici dat.

Programátorům, kteří používají jazyk pro správu dat uložených v databázi, slouží jazyk pro modifikaci dat. Jedná se o díl SQL, který umožňuje vkládání, mazaní a úpravu záznamů v relačních tabulkách.

V poslední řadě slouží jazyk pro finální prezentaci dat. Jde o jazyk pro dotazování, který je určen koncovým uživatelům. Pro formulaci dotazu využívá operace relační algebry. Tento jazyk bude vzhledem k zadání bakalářské práce dále popsán.

Hlavní rozdíl oproti běžným programovacím jazykům, jako je např. *C#*, *Python* nebo jazyk *Java* je v tom, že při použití dotazovacího jazyka neřešíme způsob, jakým jsou data získána, ale uvádíme jen informaci ohledně dat, které chceme obratem přijmout. V praxi tento proces funguje tak, že napíšeme požadavek, v tomto případě dotaz v jazyce SQL, databázový systém jej po sléze zpracuje, obstará data z databáze a vrátí je v určité formě zpět k nám.[9]

SQL tabulka je odlišná od relace nad relačním schématem. SQL tabulka je ve skutečnosti jen kolekce řádků, nemusí to být množina. Sloupce má uspořádány zleva doprava a často bývají identifikovány svojí pozicí, namísto svého názvu. Jak bylo zmíněno v kapitole 2, záhlaví, označované jako relační schéma je množina dvojic atribut-doména. Ale jelikož v SQL jsou sloupce uspořádány, bylo by

vhodnější nepovažovat záhlaví za množinu, ale za sekvenci atributů (nebo spíše sloupců).

Navíc SQL používá řádky, ne n-tice. Hodnoty v řádcích jsou uspořádány zleva doprava. Například řádek $(1, 2)$ a řádek $(2, 1)$ přestavují dva rozdílné SQL řádky.

2.2 Referenční integrita

Všechny tabulky v databázi jsou uloženy samostatně. Lze však mezi nimi vytvořit určitou návaznost a vytvořit tak v množině atributů jedné tabulky odkaz na atribut nebo skupinu atributů v tabulce další. Množina odkazovaných atributů má podobu primárních klíčů [1].

Primární klíč K tabulky R je podmnožina záhlaví tabulky R a to pouze pokud má následující vlastnosti:

- *Jedinečnost*: Žádná hodnota v R neobsahuje dvě odlišné n-tice se stejnou hodnotou na K.
- *Neredukovatelnost*: Žádná neprázdná podmnožina množiny K nesplňuje vlastnost jedinečnosti.

Počet atributů, ze kterých se skládá primární klíč, nazýváme stupeň množiny K [2].

Tabulka 3 zobrazuje modifikovanou tabulku *Student* z kapitoly 2 popisující relační model. Tabulka nyní obsahuje primární klíč na atributu *id*, který označuje každého studenta unikátním identifikačním číslem.

Referenční integrita je implementovaná pomocí cizích klíčů. Cizí klíč je množina atributů v jedné tabulce, jejichž hodnoty musí odpovídat hodnotám některých primárních klíčů v nějaké jiné tabulce.

Tabulka 3: Student - tabulka studentů s primárním a cizím klíčem

id	**jméno**	**příjmení**	**narození**	**obor_id**	**ročník**	**status studenta**
1	Petr	Novák	4.2.1998	3	1	pravda
2	Lucie	Nová	18.6.1994	1	3	nepravda
3	Josef	Němec	3.11.1997	2	2	pravda
4	Jaromír	Němec	8.4.1998	4	2	pravda
5	Lukáš	Svoboda	18.6.1995	3	2	pravda
6	Marie	Novotná	3.12.1998	3	1	pravda

Tabulka 4: Obor - tabulka univerzitních oborů

obor_ id	**katedra**	**název**	**max. kapacita**
1	Katedra fyziky a chemie	Chemie	100
2	Katedra manažerské ekonomie	Ekonomika	120
3	Katedra informatiky	Informatika	200
4	Katedra geografie	Geografie	80

V tabulce 3 - *Student* je atribut *obor_id* cizí klíč, jehož hodnoty se musí shodovat s primárním klíčem *obor_id* v tabulce 4 - *Obor*. Tabulka *Obor* reprezentuje informace o skupině univerzitních oborů, jenž mohou studenti studovat.

Jelikož tabulka *Student* obsahuje například n-tici s hodnotou 1 na atributu *obor_id*, pak tabulka *Obor* musí také obsahovat n-tici s hodnotou 1 na atributu *obor_id*. Jinak by student univerzity mohl studoval neexistující obor a databáze by tak nebyla věrným modelem reality [2].

2.3 Operace relační algebry

Pro čtení záznamů z databáze využívá relační model operace relační algebry. Každá operace relační algebry přijímá na vstupu minimálně jednu relaci nad relačním schématem a jeho výsledkem je taktéž relace. Vzhledem k tomu, že na vstupu je stejná datová struktura, jako na výstupu, umožňuje nám algebra psát vnořené relační výrazy. Tuto možnost představuje vlastnost zvaná algebraický uzávěr.

C.J.Date rozděluje ve své knize [2] operace relační algebry v relačním modelu do dvou skupin:

- Původní operace
  - projekce,
  - restrikce,
  - spojení,
  - množinové operace,
- Dodatečné a nerelační operace
  - agregace,
  - řazení,
  - a další [2].

2.3.1 Množinové operace

S ohledem na to, že relace nad relačním schématem je množina n-tic, je přirozené používat v relační algebře množinové operace [1].

Pro bližší představení jednotlivých množinových operací je v kapitole použita tabulka *Student* z kapitoly 2. Jelikož se jedná o binární operace, je nutné si zavést ještě další tabulku - *Student_2*, kterou ukazuje tabulka 5.

Tabulka 5: Student_2 - tabulka studentů

jméno	příjmení	narození	obor	ročník	status studenta
Petr	Novák	4.2.1998	Informatika	1	pravda
Karel	Svoboda	7.3.1995	Geografie	2	nepravda
Josef	Němec	3.11.1997	Ekonomika	2	pravda
Jana	Dvořáková	15.6.1993	Chemie	3	nepravda
Lukáš	Svoboda	18.6.1995	Informatika	2	pravda
Marie	Veselá	24.11.1997	Ekonomika	3	pravda

Sjednocení

Pro relace $D1$ a $D2$ nad relačním schématem R je operace sjednocení - *D1 UNION D2*, rovna relaci nad relačním schématem R. Výsledná relace je složena z n-tic t, pro které platí, že se t nachází v $D1$ nebo v $D2$ nebo v obou zadaných relacích [2].

```
1 SELECT * FROM Student
2 UNION
3 SELECT * FROM Student_2
```

Zdrojový kód 1: SQL: sjednocení

V jazyce SQL může operace sjednocení vypadat například tak, jak ukazuje zdrojový kód 1. Výsledkem tohoto dotazu je tabulka, která je ukázána v tabulce 6. Jak je vidět, do výsledku jsou zahrnuty všechny řádky z obou předaných tabulek ze vstupu.

Tabulka 6: Výsledek sjednocení

jméno	příjmení	narození	obor	ročník	status studenta
Petr	Novák	4.2.1998	Informatika	1	pravda
Lucie	Nová	18.6.1994	Chemie	3	nepravda
Josef	Němec	3.11.1997	Ekonomika	2	pravda
Jaromír	Němec	8.4.1998	Geografie	2	pravda
Lukáš	Svoboda	18.6.1995	Informatika	2	pravda
Marie	Novotná	3.12.1998	Informatika	1	pravda
Karel	Svoboda	7.3.1995	Geografie	2	nepravda
Jana	Dvořáková	15.6.1993	Chemie	3	nepravda
Marie	Veselá	24.11.1997	Ekonomika	3	pravda

Průnik

Pro relace $D1$ a $D2$ nad relačním schématem R je operace průnik - *D1 INTERSECT D2*, rovna relaci nad relačním schématem R. Výsledná relace je složena z n-tic t, pro které platí, že se t nachází v $D1$ a současně i v $D2$ [2].

```
1 SELECT * FROM Student
2 INTERSECT
3 SELECT * FROM Student_2
```

Zdrojový kód 2: SQL: průnik

Příklad množinové operace průnik ukazuje zdrojový kód 2, jehož výsledek je tabulka, která je zobrazena v tabulce 7.

Tabulka 7: Výsledek průniku

jméno	příjmení	narození	obor	ročník	status studenta
Petr	Novák	4.2.1998	Informatika	1	pravda
Josef	Němec	3.11.1997	Ekonomika	2	pravda
Lukáš	Svoboda	18.6.1995	Informatika	2	pravda

Rozdíl

Pro relace $D1$ a $D2$ nad relačním schématem R je operace rozdíl - *D1 MINUS D2*, rovna relaci nad relačním schématem R. Výsledná relace je složena z n-tic t, pro které platí, že se t nachází v $D1$ a současně se nenachází v $D2$ [2].

```
1 SELECT * FROM Student
2 EXCEPT
3 SELECT * FROM Student_2
```

Zdrojový kód 3: SQL: rozdíl

Množinový rozdíl mezi tabulkami *Student* a *Student_2* v jazyce SQL ukazuje zdrojový kód 3. Výsledná tabulka je zobrazena v tabulce 8.

Tabulka 8: Výsledek rozdílu

jméno	příjmení	narození	obor	ročník	status studenta
Lucie	Nová	18.6.1994	Chemie	3	nepravda
Jaromír	Němec	8.4.1998	Geografie	2	pravda
Marie	Novotná	3.12.1998	Informatika	1	pravda

U množinové operace rozdíl ve spojení s SQL stojí za zmínku její používání v odlišných SŘBD. MySQL nedisponuje operátory pro rozdíl a navíc ani pro

průnik, proto bývají nahrazeny složitějšími výrazy využívající spojení a vnořené dotazy. Navíc má v Oracle databázích tento operátor odlišnou podobu oproti MSSQL. Tyto odlišnosti zobrazují zdrojové kódy v příloze B.

2.3.2 Projekce

Pro relaci D nad relačním schématem R a pro relační schéma $S \subseteq R$ obsahující atributy $A, B, \ldots, C$ je projekce D na schéma S - $\pi(R)_{\{A,B,\ldots,C\}}$, rovna relaci nad relačním schématem S. Výsledná relace odpovídá množině všech takových n-tic x, pro které existuje v D n-tice t s hodnotou na atributu A, rovnající se hodnotě na A v x, s hodnotou B rovnající se hodnotě B v x, ... a s hodnotou C rovnající se hodnotě C v x.

Výsledkem projekce je tedy relace obsahující všechny n-tice, které zůstanou po odebrání určitých atributů [2].

```
1 SELECT jmeno, prijmeni
2 FROM Student;
```

Zdrojový kód 4: SQL: projekce

Projekce přes 2 atributy může vypadat tak, jak ukazuje zdrojový kód 4. Projekce je zde aplikovaná na tabulku *Student* z kapitoly Relační model 2.

Tabulka 9: Výsledek projekce

jméno	**příjmení**
Petr	Novák
Lucie	Nová
Josef	Němec
Jaromír	Němec
Lukáš	Svoboda
Marie	Novotná

Jak ukazuje tabulka 9, výsledkem projekce je nová tabulka, která zahrnuje jen předané atributy a obsahuje všechny řádky z původní tabulky.

2.3.3 Restrikce

Pro relaci D nad relačním schématem R a pro výraz typu pravdivostní hodnota θ, který může obsahovat odkazy na atributy z D, je restrikce tabulky D na základě podmínky θ relace nad relačním schématem R. Výsledná relace se skládá ze všech n-tic relace D, pro které je podmínka θ vyhodnocena na *true* [2].

```
1 SELECT *
2 FROM Student
3 WHERE rocnik = 2;
```

Zdrojový kód 5: SQL: restrikce

Pro příklad dotazu s restrikcí, který je zobrazen ve zdrojovém kódu 5 je využita tabulka *Student* z kapitoly 2 o relačním modelu. Výsledkem je tedy tabulka, která obsahuje všechny řádky, jejichž hodnoty atributu *ročník* se rovnají číslu 2. Získáme tedy všechny informace o studentech druhého ročníku. Výsledná tabulka je zobrazena v tabulce 10.

Tabulka 10: Výsledek restrikce

jméno	**příjmení**	**narození**	**obor**	**ročník**	**status studenta**
Josef	Němec	3.11.1997	Ekonomika	2	pravda
Jaromír	Němec	8.4.1998	Geografie	2	pravda
Lukáš	Svoboda	18.6.1995	Informatika	2	pravda

2.3.4 Spojení

Než se dostaneme k operaci spojení, je nutné zavést pojem spojitelnost.

Relace $D1$ a $D2$ jsou spojitelné právě tehdy, když atributy se stejnými názvy, mají stejné domény. Jinak řečeno, jsou spojitelné právě tehdy, když zřetězení relačních schémat relací $D1$ a $D2$ je samo o sobě platným relačním schématem.

Pokud jsou relace $D1$ a $D2$ spojitelné, jejich přirozené spojení (nebo jen krátce, spojení) - *D1 JOIN D2*, je relace nad relačním schématem S, které je dané zřetězením relačních schémat obou zadaných relací. Výsledná relace poté obsahuje množinu všech n-tic t, které jsou výsledkem zřetězení n-tice z $D1$ s n-ticí z $D2$.

Neformálně je výsledkem spojení množina všech kombinací n-tic z obou zadaných relací, které mají shodné hodnoty na stejnojmenných atributech [2].

```
1 SELECT jmeno, prijmeni, katedra, nazev
2 FROM Student NATURAL JOIN Obor
```

Zdrojový kód 6: SQL: přirozené spojení

Zdrojový kód 6 ukazuje příklad spojení v jazyce SQL. Jsou v něm použity tabulky *Student* a *Obor* z kapitoly 2.2 o referenční integritě. Výsledkem tohoto spojení je tabulka, která je zobrazena v tabulce 11.

Tabulka 11: Výsledek spojení

jméno	příjmení	katedra	název
Petr	Novák	Katedra informatiky	Informatika
Lucie	Nová	Katedra fyziky a chemie	Chemie
Josef	Němec	Katedra manažerské ekonomie	Ekonomika
Jaromír	Němec	Katedra geografie	Geografie
Lukáš	Svoboda	Katedra informatiky	Informatika
Marie	Novotná	Katedra informatiky	Informatika

Klíčové slovo *JOIN* představuje v SQL několik různých druhů spojení. Například operace spojení - *D1 JOIN D2 ON* θ, kde θ představuje podobně jako u restrikce 2.3.3 uživatelsky definovanou podmínku [2].

```
1 SELECT jmeno, prijmeni, katedra, nazev
2 FROM Student JOIN Obor ON Student.obor_id = Obor.obor_id
```

Zdrojový kód 7: SQL: spojení JOIN ON

Zdrojový kód 7 ukazuje příklad takového spojení. Dotaz je sestavený tak, že je jeho výsledkem stejná tabulka, jako u předchozího dotazu ve zdrojovém kódu 6, tedy tabulka 11.

2.3.5 Agregace

Agregace je v relačním modelu operace, která počítá jednu hodnotu z hodnot obsažených v zadaném atributu zadané relaci nad relačním schématem. Nebo v případě agregační funkce *COUNT*, která je od ostatních mírně odlišná, ze všech hodnot ze zadané relace. Vzhledem k tomu, že výsledkem je jediná hodnota, nikoli relace nad relačním schématem, nedá se agregace považovat za relační operaci.

Zdrojový kód 8 ukazuje výraz pro zjištění počtu řádků, které se nacházejí v tabulce *Student*, která byla definována v kapitole 2 popisující relační model.

```
1 SELECT COUNT(*) AS pocet
2 FROM Student
```

Zdrojový kód 8: SQL: agregační funkce COUNT

V jazyce SQL lze využít agregační funkce, které jsou obsaženy v následujícím seznamu, kde jsou uvedeny společně se svojí funkčností, jaké hodnoty počítají:

- *COUNT*: počet,
- *AVG*: aritmetický průměr,

- *SUM*: součet,
- *MIN*: minimum,
- *MAX:* maximum.

Jazyk SQL ve skutečnosti nepodporuje agregační operace jako takové. V SQL se tyto výrazy nevyhodnocují na samotné výpočty, a nevrací skutečně jedinou hodnotu. Vyhodnotí se na celé tabulky, které tyto výpočty obsahují. Přesněji, každý takový výraz vrací tabulku, která se skládá z jednoho řádku a jednoho sloupce. Řádek obsahuje jedinou hodnotu, což je právě vypočítaný výsledek použité agregace [2].

Jak ukazuje tabulka 12, výsledkem dotazu ze zdrojového kódu 8 ze začátku kapitoly je tedy tabulka s jedinou hodnotou ve svém těle.

Tabulka 12: Výsledek agregační funkce COUNT

pocet
6

Výrazy obsahující agregační funkce sice nepředstavují agregační operace jako takové, ale dá se říct, že jsou jakýmsi napodobením těchto operací. Ve skutečnosti je však agregace v SQL zpracovávána tak, jako by byla speciálním případem sumarizace. Pro současné účely se dá sumarizace považovat za to, co v SQL představuje výraz *SELECT* s klauzulí *GROUP BY*. Klauzule *GROUP BY* umožňuje seskupit řádky, které mají stejné hodnoty na odpovídajících atributech [2].

```
1 SELECT rocnik, MAX(narozeni) AS nejvyssi_datum
2 FROM Student
3 GROUP BY rocnik
```

Zdrojový kód 9: SQL: agregační funkce MAX

Zdrojový kód 9 ukazuje SQL výraz sumarizace s agregační funkcí počítající maximální hodnotu atributu *narození*, a to pro každý ročník. Výsledkem je proto tabulka s počtem řádků, který je roven počtu různých hodnot atributu *ročník*. Výsledek zobrazuje tabulka 13.

Tabulka 13: Výsledek agregační funkce MAX

ročník	**nejvyssi_datum**
3	18.6.1994
2	8.4.1998
1	3.12.1998

2.3.6 Řazení

Jedna z dalších užitečných operací jazyka SQL je reprezentována klauzulí *ORDER BY*. I přesto, že tato kapitola pojednává o relační algebře v relačním modelu, operace *ORDER BY* ve skutečnosti není její součástí.

Jak bylo zmíněno v kapitole 2, tabulka v první normální formě neuvažuje uspořádání svých řádků a jelikož operace *ORDER BY* slouží k seřazení řádků tabulky, postrádá tato operace v relačním modelu smysl. I přesto má v jazyce SQL smysl pro prezentaci výsledků [2].

```
1 SELECT *
2 FROM Student
3 ORDER BY narozeni
```

Zdrojový kód 10: SQL: operace ORDER BY

Zdrojový kód 10, který obsahuje odkaz na tabulku z kapitoly 2, se tedy nedá považovat za relační výraz jako takový, ale v SQL se jedná o výraz naprosto platný. Výsledkem takového výrazu by byla posloupnost řádků seřazených vzestupně podle atributu *narození*, která je zobrazena v tabulce 14.

Tabulka 14: Výsledek řazení

jméno	**příjmení**	**narození**	**obor**	**ročník**	**status studenta**
Lucie	Nová	18.6.1994	Chemie	3	nepravda
Lukáš	Svoboda	18.6.1995	Informatika	2	pravda
Josef	Němec	3.11.1997	Ekonomika	2	pravda
Petr	Novák	4.2.1998	Informatika	1	pravda
Jaromír	Němec	8.4.1998	Geografie	2	pravda
Marie	Novotná	3.12.1998	Informatika	1	pravda

3 Příprava na vývoj aplikace

Cílem této práce je vytvořit grafický nástroj pro koncové uživatele na vizuální tvorbu databázových dotazů v relačních databázových systémech. Výsledná aplikace by měla mít možnost snadným a intuitivním způsobem vytvářet a vyhodnocovat jednoduché i složitější dotazy. Navíc by neměla být závislá na konkrétním SŘBD.

3.1 Požadavky

S ohledem na cíl bakalářské práce je hlavní funkcionalita aplikace stanovena na použití jazyka pro dotazování v již existující relační databázi. K dotazování jsou využity grafické prvky tak, aby mohli aplikaci v plném rozsahu používat uživatelé s minimální nebo až nulovou znalostí dotazovacího jazyka SQL. Je možné využívat operace relační algebry, které jsou popsány v teoretické části, jako jsou projekce, restrikce, spojení, množinové operace a další.

Uživatelům je umožněno se pomocí zadaných údajů připojit k databázi, ke které mají přístup. Na výběr je ze tří typů SŘBD, na jejichž rozdílnosti nezávisí funkčnost aplikace.

Aplikace manipuluje s daty prostřednictvím jazyka SQL, což s sebou přináší nepříjemné úskalí v podobě *SQL injection*. Proto je aplikace vyvíjena s dostatečnou ochranou proti útokům tohoto typu.

3.2 Použité technologie

K vývoji aplikace byly zvoleny následující technologie:

- Microsoft C# .Net 4.6.1,
- Microsoft WPF,
- Google Material Design In XAML,
- aplikace pro správu databázových serverů:
  - Microsoft SQL Server 2017,
  - MySQL Server 8.0,
  - Oracle 12c,
- dotazovací nástroje:
  - Microsoft SQL Server Management Studio 2017,
  - MySQL 8.0 Command Line Client,
  - SQL Plus.

3.2.1 Programovací jazyk

K vývoji aplikace byl zvolen programovací jazyk *C#*. Jedná se o vysokoúrovňový objektově orientovaný programovací jazyk od společnosti Microsoft. Vytváří desktopové aplikace pro operační systém Windows. Jedná se o jazyk fungující na platformě *.NET Framework*, která je využita ve verzi 4.6.1.

Pro tvorbu uživatelského rozhraní je projekt vytvořen jako aplikace *WPF* (Windows Presentation Foundation), která je taktéž vyvinuta společností Microsoft. WPF nabízí širokou sadu funkcí a ovládacích prvků pro hladký vývoj grafické stránky aplikace. Využívá značkovací jazyk XAML, který je velice podobný jazyku HTML, tudíž je psaní statických části aplikace podobné tvorbě webu.

Uživatelské rozhraní zahrnuje využití knihoven *Material Design In XAML* od společnosti Google. Tyto knihovny jsou zdarma a volně šiřitelné a umožnily dodat moderní vzhled většině standardním ovládacím prvkům WPF.

S ohledem na zvolené programovací technologie je použito vývojové prostředí Microsoft Visual Studio ve verzi 2017.

3.2.2 Databázové systémy

Nezbytnou částí při vývoji softwaru bývá zvolení ideálního SŘBD na uchování dat programu. V případě této aplikace jde o výběr několika vhodných databázových systémů, které budou mít uživatelé aplikace k dispozici.

Pro jejich volbu bylo zvoleno jako hlavní a jediné kritérium míra používání těchto systémů ve světě. Tím je zajištěno, že bude výslednou aplikaci využívat větší množství uživatelů. S ohledem na popularitu, která je znázorněna v kapitole 1.2 zabývající se systémy řízení báze dat, byly zvoleny tyto 3 systémy:

- Oracle,
- MySQL,
- Microsoft SQL Server.

Při vývoji je nezbytné testovat funkčnost aplikace ve všech třech typech databází. K těmto účelům byl pro každý systém vytvořen lokální server pomocí odpovídajících aplikací.

Tak jak se vyvíjejí informační technologie, pokroky zaznamenává i svět databází. Jednotlivé databázové servery se mohou díky svému vývoji lišit ve svých verzích a novější verze mívají navíc větší množství funkcí. I přesto že jazyk pro dotazování je ve svých základech už několik let stejný, je nutné zmínit určitou optimalizaci vzhledem k verzím databázových serverů, na nichž mohou uživatelé pracovat. Aplikace je tedy optimalizována pro databáze, které fungují na serverech podle jejich typů. Například pro Microsoft SQL je optimální verze použití serveru *2017*, pro databáze typu MySQL je nejvhodnější použití stejnojmenného serveru s označením *8.0* a Oracle databáze je ze stejného důvodu pro verzi *12c*, někdy také označovanou *12.3*.

Při vývoji byly také využívány různé dotazovací prostředky. Jednou z těchto technologií je Microsoft SQL Server Management Studio, které má grafické uživatelské rozhraní. Tento software umožňuje mimo tvorby dotazů i správu jakékoli infrastruktury SQL, včetně správy serveru. Další z dotazovacích nástrojů je MySQL Command Client pro řízení dotazů v MySQL databázi a také SQL Plus, která dokáže obsluhovat relační databáze Oracle. Oba tyto databázové prostředky využívají pro svou práci příkazový řádek, což jim neubírá na funkcionalitě.

4 Postup při vývoji

Po určení všech technologií a specifikací na výslednou aplikaci, přichází řada na její samotnou tvorbu.

4.1 Databázové spojení

Vzhledem k tomu, že dle specifik má aplikace nabízet tři rozdílné SŘBD, bylo nutné obstarat knihovny, které toto spojení zprostředkovávají.

Pro jednotlivé systémy řízení byly vytvořeny samostatné třídy, které obsahují metody pro manipulaci s danou databází, jako například:

- navázání a ukončení spojení,
- vrácení názvů tabulek a sloupců,
- zjištění primárních a cizích klíčů,
- vyhodnocení dotazu
- a další.

4.2 Skládání dotazu

Každou tabulku reprezentuje instance třídy *Table*. Objekt typu *Table* má zapouzdřené mimo jiné informace o svém názvu, aliasu a o svých sloupcích.

Pro tabulky, sloupce a jednotlivé operace jsou v aplikační vrstvě připraveny datové struktury ve formě generických kolekcí. Při přidání sloupce, tabulky nebo při použití nějaké z nabízených operací se před přesnou formulací dotazu vloží použitá operace do příslušného seznamu. Následně se algoritmem složí obsahy všech seznamů do statické proměnné typu *string*, která reprezentuje SQL dotaz. S ohledem na aktuálně připojený typ databáze se zavolá metoda pro vyhodnocení dotazu a při úspěchu se výsledek zobrazí v prezentační vrstvě aplikace.

4.3 Ochrana proti útokům

Aplikace je určena pro usnadnění základní práce v SQL databázích, proto je více než vhodné, aby byla chráněna proti útokům SQL injection.

K ochraně byla použita nejbezpečnější metoda využívající parametrizované dotazy. Dotaz je připravován tak, že namísto veškerých hodnot, které mohou být zadány uživatelem, jsou dosazeny zástupné znaky. A následně jsou samotné hodnoty předány odděleně od dotazu. Textový řetězec, který představuje v pozadí aplikace odesílaný dotaz, může být odeslán například v tomto tvaru:

SELECT * FROM Student WHERE jméno = @0 AND příjmení = @1;

V aplikační části jsou k těmto účelům připraveny speciální metody. Jedna z nich je určena pro vyhodnocení dotazu v databázovém systému na serveru typu Microsoft SQL. A je zobrazena v podobě zdrojového kódu v příloze A.

5 Uživatelská příručka

Aplikace je určena především uživatelům, kteří pracují s relační databází a potřebují z ní získat data ve formě tabulek. Uživatelům je umožněno vytvářet dotazy a zobrazovat jejich výsledky bez použití dotazovacího jazyka SQL. To vše pomocí grafických ovládacích prvků, na jejichž intuitivní použití byly kladeny vysoké nároky. Z tohoto důvodu by ovládání aplikace neměl být problém i pro méně zkušeného uživatele.

K jejímu spuštění není vyžadována instalace a stačí spustit soubor *VisualQueryBuilder.exe*, který se nachází na přiloženém CD.

K účelům testování se v přiloženém souboru nachází připojovací řetězec pro připojení k testovací databázi.

5.1 Připojení k databázi

Po spuštění aplikace je k dispozici okno, které slouží pro přihlášení uživatele a spojení s databází. Bez připojení k některé databázi není možné aplikaci využívat.

Obrázek 2: Okno pro připojení k databázi

Okno, které je zobrazeno na obrázku 2, obsahuje mimo jiné tab menu. Toto menu je tvořeno třemi nabízenými SŘBD, se kterými může uživatel pracovat.

Pro připojení k vlastní databázi je třeba si jeden ze systémů zvolit a vyplnit požadované údaje.

Pro připojení k databázi je možné kromě vyplnění databázových údajů použít speciální připojovací řetězec. K tomuto účelu slouží tlačítko *použít připojovací řetězec*. Po kliknutí na tlačítko se změní podoba okna, kterou ukazuje obrázek 3.

Obrázek 3: Okno pro připojení k databázi pomoci připojovacího řetězce

Zmiňovaný řetězec je třeba překopírovat do určeného pole a následně potvrdit připojení tlačítkem *PŘIPOJIT*. Pro návrat k možnosti vyplňování databázových údajů slouží tlačítko *ZRUŠIT*.

Aplikace navíc umožňuje zapamatovat si aktuálně vyplněné údaje pro opakované použití. K zapamatování stačí zaškrtnout pole *Zapamatovat*, které se nachází jako nejspodněji umístěný prvek připojovacího okna.

5.2 Změna jazyka

Výsledná aplikace je primárně v anglickém jazyce, je však plně lokalizovaná do češtiny.

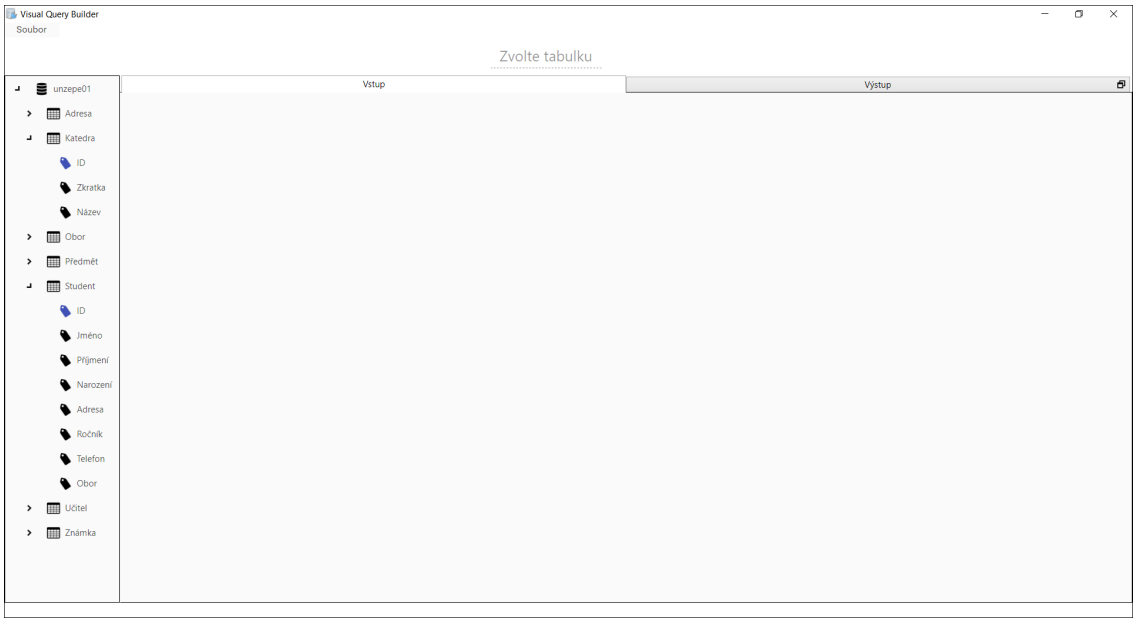
K volbě jazyka dochází při připojení k databázi, tedy v okně, které se zobrazí při spuštění aplikace. Toto okno je ukázáno na obrázku číslo 2. K výběru jazyka slouží nabízené vlaječky, jejichž kliknutím lze jazyk zvolit.

Úspěšnou změnu je možné sledovat už v aktuálním okně, jehož prvky se po kliknutí na vlajku *přeloží* a změní tak podobu svého obsahu.

Navíc si aplikace po úspěšném připojení pamatuje zvolený jazyk, a není tedy nutné při dalším použití aplikace jazyk znovu volit.

5.3 Hlavní okno

Po databázovém připojení má uživatel k dispozici hlavní okno aplikace. V tomto okně se vytváří dotazy a je v něm reprezentována veškerá funkčnost programu. Okno může vypadat tak, jak ukazuje obrázek 4.



Obrázek 4: Hlavní okno aplikace

5.3.1 Menu

Menu umístěné v levém horním rohu představuje jediná položka, která má typicky název *Soubor* a obsahuje další položky.

Podpoložka pojmenovaná *Nové spojení* slouží k vytvoření nového spojení s databází. Lze jí využít pro obnovení spojení s aktuální databází nebo k připojení k nové databázi.

Položka *Konec* ukončí aktuálně vytvořené spojení a zavře aplikaci.

5.3.2 Seznam tabulek se sloupci

Rozbalovací menu, které se nachází po skoro celé výšce v levé části okna je určeno pro výpis všech tabulek, které se v připojené databázi nacházejí. Obsahuje názvy tabulek společně s jejich sloupci.

První položka tohoto menu s ikonou představující databázi (🗄) označuje název aktuálně připojené databáze. Její přímí potomci jsou položky, které reprezentují jednotlivé tabulky. Ty jsou také označeny příhodnými ikonami (▦). Sloupce konkrétních tabulek jsou označeny těmito ikonami: 🏷. Pokud je daný sloupec primárním klíčem je odlišen od ostatních změnou barvy (🏷).

5.3.3 Vstup a výstup

Zbytek okna slouží pro přesnou formulaci dotazů a pro zobrazení jejích výsledků. Tento prostor je tvořen tab menu, které se skládá ze dvou karet.

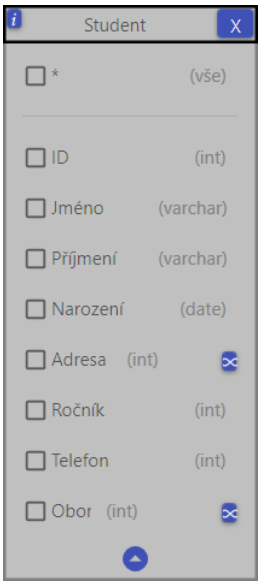
Karta označená jako *Vstup* je určena pro zobrazení jednotlivých tabulek a prostřednictvím grafické interakce umožňuje skládat dotazy. Pro bližší informace k reprezentaci tabulek a vytváření dotazů slouží podkapitola 5.4 s názvem Tabulka.

Po úspěšném vytvoření dotazu je vhodné zobrazit jeho výsledky. Výsledky jsou k dispozici ve formě tabulky na kartě *Výstup*. Tuto kartu je možné si přizpůsobit, což ocení zejména majitelé větších monitorů. K těmto účelům je hlavička karty vybavena ikonou (🗗). Po jejím kliknutí se umístí obsahy obou karet pod sebe, což vyplní nevyužité místo a předejde se nadbytečnému přepínání mezi kartami.

Tento proces lze samozřejmě vrátit zpět, k čemuž slouží ikona, jejichž podoba je v tomto režimu nahrazena ikonou maximalizace (🗖).

5.4 Tabulka

V aplikaci jsou tabulky reprezentovány jako grafické objekty, které zobrazují struktury konkrétních tabulek. Tyto objekty jsou v aplikaci i příručce označovány jako samotné tabulky. Do dotazu je lze zahrnout jejich vložením do obsahu karty *Vstup*. Prostřednictvím dvojkliku na daný název tabulky ze seznamu tabulek se sloupci lze toto vložení uskutečnit.



Obrázek 5: Reprezentace tabulky v aplikaci

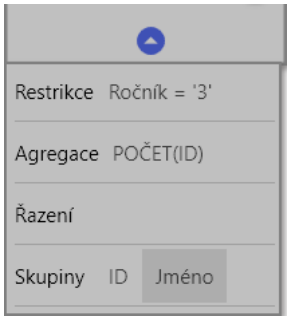
Na obrázku 5, je zobrazen možný vzhled tabulky z aplikace. Je možné si přizpůsobit její pozici v okně. Stačí jí uchytit levým tlačítkem myši a přetáhnout na zvolené místo.

Pro odstranění tabulky z karty a její vyjmutí z dotazu slouží tlačítko zavření. Je umístěno v horním pravém rohu a má klasickou podobu křížku.

5.4.1 Použité operace

Tlačítko zobrazené v levém rohu tabulky, které má podobu informační ikony (***i***) slouží k zobrazení či schování použitých operací. Zobrazí obsah, který je určen k manipulaci s použitými operacemi, které souvislý jen a výhradně s danou tabulkou. Jak vytvářet veškeré operace je podrobně popsáno v podkapitole 5.5 s názvem Operace.

Po kliknutí na zmiňované tlačítko lze v seznamu pod tabulkou nalézt použité restrikce, aplikované agregační funkce, řazení výsledků a vytvořené skupiny spojené se sloupci tabulky. Seznam je zobrazen na obrázku 6.



Obrázek 6: Zobrazení použitých operací na tabulce

Řádek se skupinami slouží jen k informativním účelům a nelze s jednotlivými položkami pracovat. Je tomu tak kvůli zachování korektnosti dotazu a odvíjí se od zvolených sloupců vzhledem k použité agregaci.

Ostatní položky je možné pravým tlačítkem a adekvátní volby odebrat. Odebrání vyjme danou operaci z dotazu, což se okamžitě projeví ve výsledku, tedy na kartě ıVýstup.

5.4.2 Přejmenování tabulky

Může se stát, že bude mít uživatel zobrazených více stejných tabulek. Bez jejich manuálního přejmenování by všechny tyto tabulky měly stejné názvy a odlišit by je bylo možné jen pořadovým číslem. Celý název tabulky i s jeho pořadovým číslem je zobrazen v její hlavičce, která je zobrazena na obrázku 5.

Kliknutí pravým tlačítkem na hlavičku tabulky a následné volby pro přejmenování lze tabulku přejmenovat. Následně je nutné vyplnit políčko pro název a potvrdit změnu klávesou *enter*. Název tabulky však musí splňovat následující pravidla:

- nesmí být stejný, jako už existující tabulka na kartě,
- musí začínat písmenem,

- smí obsahovat pouze písmena, číslice nebo speciální znak podtržítka.

Podtržítko je vhodné zejména pro označení mezery ve víceslovných názvech.

5.4.3 Sloupce

Tabulka je tvořena zejména seznamem jeho sloupců. Položky tohoto seznamu se skládají ze zaškrtávacího políčka, názvu sloupce, názvu jeho datového typu a případně i tlačítka s ikonou značící propojení (⋊), která značí výskyt cizího klíče na daném sloupci.

Zaškrtnutím zmiňovaného zaškrtávacího políčka lze vložit daný sloupec do dotazu a zahrnout tak jeho hodnoty do výsledné tabulky. Opětovné kliknutí na políčko, tedy odstranění jeho zaškrtnutí, daný sloupec z dotazu naopak vyjme.

Aplikace umožňuje určit pořadí jednotlivých sloupců a seřadit je tak ve výsledku. Tuto funkčnost lze provést s využitím stisknutého levého tlačítka myši a následného táhnutí přes jednotlivé položky, které představují samotné sloupce. Tímto způsobem se změní pozice sloupců a po puštění tlačítka myši se seřadí i ve výsledku.

Pro zjištění odkazu cizího klíče, stačí najet myší na tlačítko k tomu určené. Zobrazí se text, jenž značí název tabulky s názvem odkazovaného sloupce, které jsou odděleny tečkou.

Pokud je potřeba tabulku s odkazovaným sloupcem zobrazit a zahrnout jí tak do dotazu, není nutné ji hledat v seznamu tabulek se sloupci. Lze využít zmiňované tlačítko a jednoduchým dvojklikem zobrazit tabulku.

5.4.4 Přejmenování sloupce

Často se stává, že uživatel pracuje s databází, kde je potřeba si přizpůsobit názvy sloupců. Ať už je to z důvodu nevhodně, nebo nedostatečně pojmenovaných sloupců nebo pro lepší organizaci, lze si sloupec po dobu dotazování přejmenovat podle sebe. K těmto účelů stačí kliknout levým tlačítkem na daný sloupec a vyčkat, než se zpřístupní políčko pro zvolení názvu. Vyplnit název a potvrdit klávesou *enter*.

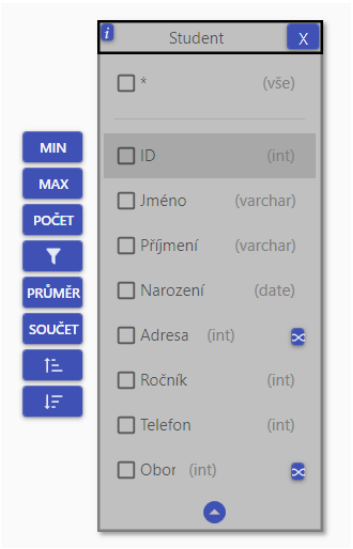
Je nezbytné brát v potaz, že pro názvy sloupců platí stejná pravidla, jako pro tabulky.

5.5 Operace

Aplikace umožňuje vytvářet operace jejich postupným vkládání do dotazu. Použití je rozděleno podle toho, zda se jedná o operace týkající se sloupce jedné tabulky nebo jde o více tabulkové operace.

5.5.1 Operace na jedné tabulce

Pro vytvoření operací tohoto typu je k dispozici lišta, která nabízí operace v podobě tlačítek. Je možné ji zobrazit kliknutím pravým tlačítkem myši na položku sloupce v tabulce.



Obrázek 7: Nabízené operace na tabulce

Po kliknutí se lišta objeví po levé straně tabulky tak, jak je ukázáno na obrázku 7. Počet operací se mění v závislosti na datovém typu daného sloupce. Pro sloupec, který je textového typu, jako například *varchar* nebo *char*, není umožněna aplikace agregačních funkcí počítajících průměr nebo součet.

Každé tlačítko reprezentuje jinou databázovou operaci. V následující tabulce 15 jsou uvedeny všechny tlačítka a jaké operace představují.

Tabulka 15: Operace na jedné tabulce

Tlačítko	**Funkčnost**	**SQL**
MIN	Vrací minimální hodnotu ve sloupci	MIN()
MAX	Vrací maximální hodnotu ve sloupci	MAX()
POČET	Vrací počet hodnot ve sloupci	COUNT()
	Omezí hodnoty ve sloupci	WHERE
PRŮMĚR	Vypočítá průměr hodnot ve sloupci	AVG()
SOUČET	Sečte hodnoty ve sloupci	SUM()
	Seřadí hodnoty vzestupně podle daného sloupce	ORDER BY ASC
	Seřadí hodnoty sestupně podle daného sloupce	ORDER BY DESC

Pro provedení operace a její zahrnutí do dotazu je nutné kliknout na odpovídající tlačítko.

Pokud se jedná o omezení přípustných hodnot pomocí restrikce, zobrazí se po kliknutí na pravé straně tabulky ve výšce zvolené položky grafický prvek. Tento prvek vyžaduje další interakci, kterou lze úspěšně dokončit stisknutím klávesy *enter*. Zavřít jí lze pomocí klávesy *esc* nebo kliknutím do jiného místa v okně.

Tento prvek se skládá z názvu slupce, aritmetického operátoru pro porovnání hodnot a textového pole, které slouží k manuálnímu zadání porovnávající hodnoty. Znak porovnání je nutné vybrat z poskytnutého rozbalovacího menu a hodnotu pro porovnání je možné napsat ručně nebo zvolit z aktuální tabulky. K tomuto účelu slouží tlačítko, které se nachází na pravém konci celého prvku. Po jeho kliknutí se zobrazí všechny hodnoty v dané tabulce seřazeny do sloupců a řádků. Jedním kliknutím na požadovaný řádek se zvolí hodnota daného sloupce do textového pole. Dvojklikem se navíc restrikce potvrdí a dosadí se tak do dotazu.

Pokud vstupní data aplikace obsahují více restrikcí v daný moment, musí výsledné řádky splňovat všechny zadané podmínky současně.

Při opakovaném aplikování restrikce na stejný sloupec jsou však ve výsledku zobrazeny řádky, u kterých stačí, když je splněna jen jedna ze zadaných podmínek. Tím se omezí nechtěným prázdným výsledkům.

Po úspěšném vykonání jedné z operací se ihned složí dotaz a vyhodnotí se. Bezprostředně po vyhodnocení se zobrazí výsledky na odpovídající kartě *Výstup*.

Je důležité brát v potaz, že výstupní struktura dokáže obsáhnout maximálně 1000 výsledků. Proto je někdy vhodné použít restrikci, pro zkonkretizování požadovaných dat a omezení tak počtů záznamů ve výsledku.

5.5.2 Více tabulkové operace

Pojmem více tabulkové operace jsou označeny operace, pro jejichž provedení je nutná účast dvou a více tabulek. Aplikace nabízí následující operace tohoto typu:

- množinové operace mezi dvěma tabulkami,
- vnitřní spojení na rovnost.

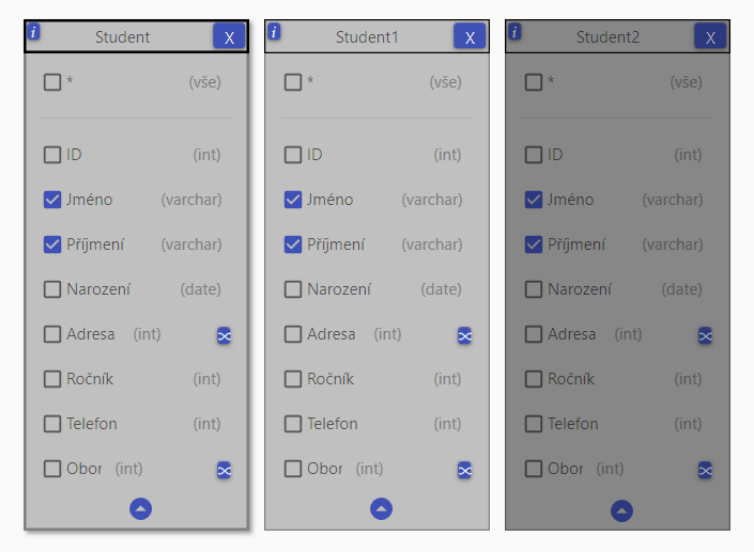
Nabízené operace lze zvolit a použít z horizontální lišty, která se zobrazí po vybrání dvou odpovídajících tabulek.

Více tabulek lze vybrat dvěma způsoby. Jeden ze způsobů, jak vybrat více tabulek je pomocí klávesy *ctrl*. Držením této klávesy a postupným klikáním na jednotlivé tabulky lze vybrat jednu po druhé. Aktuálně vybraná tabulka má zvýrazněné okraje, a navíc obsahuje vnější stín.

Dalším způsobem je použití techniky *táhni a pusť*. Nejdřív je nutné stisknout a držet levé tlačítko myši, a to kdekoli na volné ploše v kartě *Vstup*. Následným táhnutím lze automaticky zvětšovat vytvořený obdélník, do jehož obsahu je potřeba zahrnout tabulky, které je třeba vybrat. Pro vybrání tabulky stačí, aby se

ve zmiňovaném obdélníku objevila část její hlavičky. Pro potvrzení výběru stačí pustit tlačítko myši.

Korektní vybrání je určeno pouze pro dvě tabulky v danou chvíli. Ostatní tabulky se posunou do pozadí a dočasně se vyjmou z dotazu. Odsunuté tabulky lze poznat podle jejich zašednutí.



Obrázek 8: Rozdíl mezi aktuálně vybranou, běžnou a nepoužitelnou tabulkou

Obrázek 8 se skládá ze tří tabulek. První zleva je aktuálně vybraná a má daný vzhled podle zmiňovaného popisu. Navíc má zobrazený svůj název ve vrchní části uprostřed celého okna. Další tabulka je běžná, není momentálně vybraná a veškeré tabulkové operace i vybrané sloupce se objeví ve výsledku dotazu. Poslední je tabulka, která má tmavší vzhled, což značí její odsunutí.

Jakmile jsou vybrány jedním ze způsobů dvě tabulky, objeví se v levé horní části hlavního okna lišta s tlačítky. Ta má velice podobný vzhled, jako lišta pro jedno tabulkové operace. Na obrázku 9 je zobrazena lišta pro operace mezi více tabulkami.

Obrázek 9: Nabízené operace mezi dvěma tabulkami

Jednotlivá tlačítka představují více tabulkové operace a jsou vysvětleny v tabulce 16.

Tabulka 16: Více tabulkové operace

Tlačítko	Funkčnost	SQL
	Vytvoří sjednocení mezi tabulkami	UNION
	Vytvoří průnik mezi tabulkami	INTERSECTION
	Vytvoří rozdíl mezi tabulkami	EXCEPT
	Umožní vytvořit spojení mezi tabulkami	JOIN
	Odstraní množinovou operaci a skryje lištu	

5.5.3 Spojení

Předposlední tlačítko z lišty více tabulkových operací slouží pro vytvoření spojení na sloupcích z vybraných tabulek. Tlačítko obsahuje ikonu spojení, podobné znaku nekonečna. Po jeho kliknutí je uživateli k dispozici dialogové okno, které je zobrazeno na obrázku 10.

Obrázek 10: Dialogové okno pro spojení

Okno se skládá ze tří vertikálních seznamů a několika tlačítek. Seznamy, které se vyskytují na obou koncích okna reprezentují tabulky se sloupci. Sloupce lze pomocí dvojkliku nebo s využitím tlačítka se znakem sčítání (+) přesunout do střední části okna. Zde se nacházejí vybrané sloupce z obou tabulek a lze je odebrat z tohoto seznamu také prostřednictvím dvojkliku nebo s využitím tlačítka se znakem odčítání (-). Sloupce, které se nacházejí v jednom řádku v seznamu vedle sebe budou ve výsledku zahrnuty do operace spojení a budou spojeny na základě spojení na rovnost. Střední seznam musí obsahovat stejný počet položek na obou stranách a navíc musí být datové typy mezi sebou porovnatelné. Pokud nebudou tyto podmínky splněny, spojení se stane neuskutečnitelné a nebude možné operaci potvrdit. Navíc bude uživatel upozorněn výrazným textem v levé spodní části dialogového okna.

Pro vyprázdnění středního seznamu slouží tlačítko s textem odebrat vše. Tlačítko obsahující slovo zrušit provede vrácení momentálně vytvořených změn a za-

vře dialogové okno. Po provedení operace prostřednictvím adekvátního tlačítka, se spojení zařadí do dotazu, který se vyhodnotí a aktuální dialogové okno se zavře. Navíc se pro zachování korektnosti dotazu odstraní množinová operace, pokud je momentálně využita.

Spojení jsou reprezentované čárou mezi tabulkami s odpovídající ikonou uprostřed. Navíc mají možnost pro opětovné zobrazení dialogového okna a lze tak odstranit nebo upravit jednotlivá spojení. Tuto funkčnost je možné vyvolat dvojklikem na ikonu, která se nachází na zmiňované čáře mezi tabulkami.

5.5.4 Množinové operace

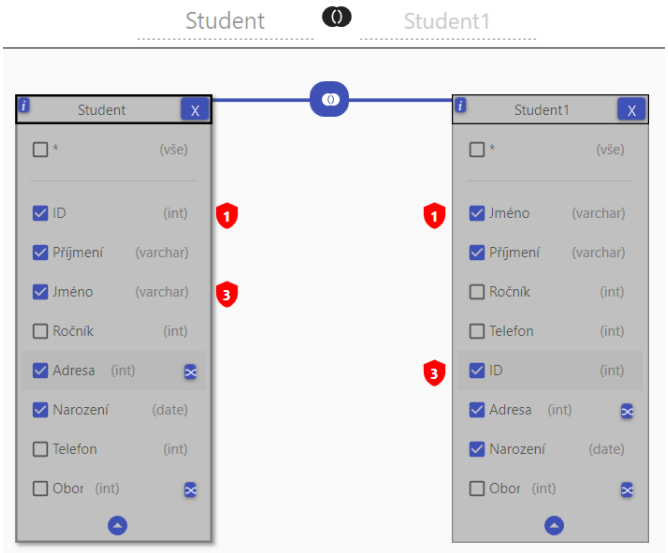
První tři tlačítka z lišty pro operace mezi dvěma tabulkami, která je zobrazena na obrázku 9 představují množinové operace a vyhodnotí se ihned po jejich kliknutí.

Jejich aplikování nenávratně upraví výsledný dotaz a omezí funkčnost aplikace po dobu využití množinových operací. Změny se týkají následujícího:

- odstranění všech spojení mezi tabulkami,
- odstranění veškerého třídění výsledků,
- znemožnění třídění výsledků.

Všechny tyto změny jsou pro zachování správnosti dotazu a bylo by velmi komplikované zahrnout je do výsledku.

Operace je podobně jako spojení reprezentovaná čárou mezi zvolenými tabulkami s danou ikonou v jejím středu. Navíc je uvedena uprostřed horní části okna, kde jde vidět pořadí tabulek v dotazu. Pořadí je důležité zejména pro operaci rozdíl.



Obrázek 11: Ukázka množinové operace

Na obrázku číslo 11 jsou uvedeny tabulky, na které je aplikován průnik. Navíc obě obsahují červené štítky s čísly, které značí neproveditelnost dotazu. Štítky označují sloupce z obou tabulek, které jsou mezi sebou neporovnatelné. Čísla v nich znamenají pozice, na kterých se jednotlivé sloupce porovnávají.

Pokud štítek obsahuje místo čísla vykřičník a vyskytuje se na úrovni položky pro volbu všech sloupů, může to znamenat jednu z možností. Buď není vybraný stejný počet sloupců, nebo je odlišný počet agregačních funkcí, který musí být stejný v obou tabulkách. Všechny tyto informace jsou dostupné po najetí myší na jednotlivé štítky.

Množinová operace proběhne jen v případě, že jsou všechny tyto chyby eliminovány a tabulky neobsahují ani jeden štítek. V opačném případě zůstane část okna pro výsledky prázdná.

Jakákoliv interakce, která nesouvisí s tabulkami v množinové operaci nebo změnou samotné operace, ukončí aktuálně použitou množinovou operaci a vyjme jí z dotazu. Pro tyto účely však slouží primárně tlačítko ZRUŠIT nacházející se v liště.

Závěr

Výsledkem této práce je funkční aplikace určená koncovým uživatelům pro tvorbu dotazů v relačních databázových systémech.

K vytvoření výsledné aplikace vedla skutečnost, že databázové systémy využívají pro prezentaci dat i uživatelé bez zkušenosti s dotazovacím jazykem SQL. Proto umožňuje snadným a intuitivním způsobem vytvářet a vyhodnocovat dotazy s minimální nebo až nulovou znalostí dotazovacího jazyka.

Vzhledem ke stanoveným cílům bakalářské práce, umožňuje aplikace vytvářet operace zahrnující projekci, restrikci, spojení, množinové operace, agregaci a další. Navíc je schopna pracovat ve třech typech SŘBD.

V teoretické části bakalářské práce je popsána problematika zabývající se relačními databázovými systémy, relačním modelem dat a způsobem pro jeho manipulaci, který využívá operace z relační algebry. Nechybí ani základy jazyka SQL, pro detailnější pochopení funkčnosti aplikace.

V praktické části práce je uvedena příprava a postup vývoje samotného programu. Také je vysvětleno správné používání aplikace v uživatelské příručce.

Aplikace využívá prvky jazyka SQL, které jsou určeny hlavně pro koncové uživatele, ale dala by se dále rozšířit o více elementů, které jazyk nabízí. Jednalo by se o části jazyka určeného pro definici a modifikaci dat.

Conclusions

The result of this work is a functional application designed to create queries in a relational database system for end users.

The fact that database systems are used for the data presentation even by users without experience with the SQL query language led to create the application. Therefore, it allows easily and intuitively create and evaluate queries with minimal or even zero knowledge of the query language.

Due to the set goals of the bachelor thesis, the application allows to create operations including projection, restriction, join, set operations, aggregation and even more. In addition, it is able to work in three types of DBMS.

The theoretical part of the bachelor thesis describes the problems dealing with the relational database systems, relational data model and method for its manipulation, which uses operations from relational algebra. There are also basics of SQL language for more detailed understanding of the application functionality.

The practical part of the work presents the preparation and development process of the program itself. The correct use of the application in the user guide is there also explained.

The application uses elements of the SQL language, which are intended mainly for end users. However it could be further extended by more elements which the language offers. It would be part of the language designed to define and modify data.

A Ochrana proti SQL Injection

```
1  public override DataTable QueryOutput(string query,
       Dictionary<string, string> parameters)
2          {
3             DataTable dt = new DataTable();
4             using (cmd = new SqlCommand(query, conn))
5             {
6                parameters.ToList().ForEach(x => cmd.Parameters.
                     AddWithValue(x.Key, x.Value));
7                SqlDataAdapter da = new SqlDataAdapter(cmd);
8                da.Fill(dt);
9             }
10            return dt;
11         }
```

Zdrojový kód 11: C#: metoda pro vyhodnocení parametrizovaného dotazu na MSSQL

B Množinová operace rozdíl v odlišných SŘBD

```
1  SELECT _A.jmeno AS jmeno, _A.prijmeni AS prijmeni
2  FROM Student _A
3  EXCEPT
4  SELECT _B.jmeno AS jmeno, _B.prijmeni AS prijmeni
5  FROM Student_2 _B;
```

Zdrojový kód 12: SQL: Množinová operace rozdíl v Microst SQL Server

```
1  SELECT xA.jmeno AS jmeno, xA.prijmeni AS prijmeni
2  FROM Student xA
3  MINUS
4  SELECT xB.jmeno AS jmeno, xB.prijmeni AS prijmeni
5  FROM Student_2 xB;
```

Zdrojový kód 13: SQL: Množinová operace rozdíl v Oracle

```
1 SELECT DISTINCT _A.jmeno AS jmeno, _A.prijmeni AS prijmeni
2 FROM Student _A
3 WHERE (_A.jmeno, _A.prijmeni) NOT IN
4 (SELECT _B.jmeno, _B.prijmeni FROM Student_2 _B);
```

Zdrojový kód 14: SQL: Množinová operace rozdíl v MySQL

C Obsah přiloženého CD/DVD

Na samotném konci textu práce je uveden stručný popis obsahu přiloženého CD/DVD, tj. jeho závazné adresářové struktury, důležitých souborů apod.

bin/
: Program VisualQueryBuilder, spustitelný přímo z CD/DVD. Adresář obsahuje i všechny runtime knihovny a další soubory potřebné pro bezproblémový běh programu z CD/DVD.

doc/
: Text práce ve formátu PDF, vytvořený s použitím závazného stylu KI PřF UP v Olomouci pro závěrečné práce, včetně všech příloh, a všechny soubory potřebné pro bezproblémové vygenerování PDF dokumentu textu (v ZIP archivu), tj. zdrojový text textu, vložené obrázky, apod.

src/
: Kompletní zdrojové texty programu VisualQueryBuilder se všemi potřebnými (příp. převzatými) zdrojovými texty, knihovnami a dalšími soubory potřebnými pro bezproblémové vytvoření spustitelných verzí programu.

readme.txt
: Instrukce pro spuštění programu VisualQueryBuilder, včetně všech požadavků pro jeho bezproblémový provoz. Popisuje také návod, jak se připojit k testovací databázi.

Odkazy

[1] Richard L. Tenney Dan A. Simovici. *Relational Database Systems*. Academic Press, 1995. ISBN: 0-12-644375-0.

[2] C.J. Date. *SQL and Relational Theory: How to Write Accurate SQL Code*. O'Reilly Media, Inc., 2011. ISBN: 978-1-449-31640-2.

[3] db-engines.com. *DB-Engines Ranking*. 2020. URL: https://db-engines.com/en/ranking.

[4] James R. Groff. *SQL: Kompletní průvodce*. Computer Press, 2005. ISBN: 80-251-0369-2.

[5] Laurenčík Marek. *SQL: podrobný průvodce uživatele*. Grada Publishing, 2018. ISBN: 978-80-271-0774-2.

[6] Krajča Petr. *Databázové systémy*. Univerzita Palackeho v Olomouci. 2019.

[7] Robert D. Schneider. *MySQL: oficiální průvodce tvorbou, správou a laděním databází*. Grada Publishing, 2006. ISBN: 978-80-247-1516-2.

[8] Jaromír Skřivan. *Datové modely a návrhy relačních schémat*. 2008.

[9] Jana Šarmanová. *Teorie zpracování dat*. 2007.

[10] Jan Tyrychtr. *Provozní a analytické databáze: Teoretické základy*. Česká společnost pro vzdělání a inovace v zemědělství, 2015. ISBN: 978-80-879-6802-4.

[11] Kucherková Zdeňka. *Úvod do databázových systémů*. Obchodní akademie Orlová, 2006. ISBN: 978-80-87113-24-0.