Česká zemědělská univerzita v Praze

Provozně ekonomická fakulta

Katedra informačního inženýrství

Bakalářská práce

Návrh relační databáze pro správu odepsaných klientů

Jan Franěk

ČESKÁ ZEMĚDĚLSKÁ UNIVERZITA V PRAZE

Provozně ekonomická fakulta

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

Jan Franěk

Informatika

Název práce

Návrh relační databáze pro správu odepsaných klientů

Název anglicky

Design of relational database for managing written-off customers

---

Cíle práce

Hlavní cíl

Navrhnout funkční model relační databáze pomocí jazyka SQL, který bude sloužit jakožto vzor pro možnou implementaci do vymáhacích či nebankovních společností.

Dílčí cíle

- Analyzovat potřebné informace a současný stav správy a vymáhání odepsaných klientů.
- Vytvořit testovací prostředí s daty.
- Ověřit funkčnost návrhu na testovací databázi.

Metodika

Metodika řešení teoretické části bakalářské práce bude založena na studiu a analýze odborných informačních zdrojů a informací z praxe. Na základě znalostí získaných v teoretické části práce budou v praktické části identifikovány vhodné postupy a rozložení databázové struktury. Na základě syntézy teoretických poznatků a výsledků praktické časti budou formulovány závěry práce.

Doporučený rozsah práce

30-40

Klíčová slova

relační databáze, SQL, databázový model, vymáhání, odpis, insolvence, zesplatnění úvěru

---

Doporučené zdroje informací

HOTEK, M. – Microsoft SQL Server 2008: krok za krokem. Brno: Computer Press, 2009, 488 s. ISBN 978-80-251-2466-6

MERUNKA, Vojtěch; VOSTROVSKÝ, Václav. *Databázové systémy.* Praha: Credit, 1998. ISBN 80-213-0388-3.

MOLLINARO, A. – SQL: kuchařka programátora. Brno: Computer Press, 2009, 573 s. ISBN 978-80-251-2617-2.

POKORNÝ, Jaroslav. *Databázové systémy a jejich použití v informačních systémech.* Praha: Academia, 1992. ISBN 80-200-0177-8.

SOUKUP, Ron; KRÁSENSKÝ, David. *Mistrovství v SQL serveru 6.5: podrobný průvodce návrhem, implementací a optimalizací databází a architekturou SQL serverů.* Praha: Computer Press, 1998. ISBN 80-7226-092-8.

VOSTROVSKÝ, Václav; ČESKÁ ZEMĚDĚLSKÁ UNIVERZITA V PRAZE. KATEDRA INFORMAČNÍHO INŽENÝRSTVÍ. *Vytváření databází v ORACLE.* V Praze: Česká zemědělská univerzita, Provozně ekonomická fakulta, 2014. ISBN 978-80-213-1191-6.

Předběžný termín obhajoby

2022/23 LS – PEF

Vedoucí práce

Ing. Martin Pelikán, Ph.D.

Garantující pracoviště

Katedra informačního inženýrství

Elektronicky schváleno dne 27. 2. 2023

Ing. Martin Pelikán, Ph.D.

Vedoucí katedry

Elektronicky schváleno dne 2. 3. 2023

doc. Ing. Tomáš Šubrt, Ph.D.

Děkan

V Praze dne 28. 01. 2024

Čestné prohlášení

Prohlašuji, že svou bakalářskou práci "Návrh relační databáze pro správu odepsaných klientů" jsem vypracoval samostatně pod vedením vedoucího bakalářské práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou v práci citovány a uvedeny v seznamu použitých zdrojů. Jako autor uvedené bakalářské práce dále prohlašuji, že jsem v souvislosti s jejím vytvořením neporušil autorská práva třetích osob.

V Praze dne 29. 01. 2024

Poděkování

Rád bych touto cestou poděkoval Ing. Martinu Pelikánovi, Ph.D. za odborné vedení, podporu, pochopení a cennou výpomoc, kterou poskytl při psaní této práce. Dále děkuji svým rodinným příslušníkům za projevenou podporu a korektury. Též děkuji svému zaměstnavateli za ochotně poskytnutý čas a zkušenosti s problematikou.

Návrh relační databáze pro správu odepsaných klientů

Abstrakt

Metodika řešení teoretické části bakalářské práce bude založena na studiu a analýze odborných informačních zdrojů a informací z praxe. Na základě znalostí získaných v teoretické části práce budou v praktické části identifikovány vhodné postupy a rozložení databázové struktury. Na základě syntézy teoretických poznatků a výsledků praktické časti budou formulovány závěry práce.

V teoretické části se práce zaměřuje na obecnou problematiku odpisů a jejich vzniku a metodiku návrhu relační databáze za uvedeným účelem.

Praktická část si klade za cíl vytvořit návrh databáze pro správu odpisových složek a klientů, za účelem možnosti implementace tohoto návrhu do standardního pracovního procesu vymáhacích a nebankovních společností.

Klíčová slova: relační databáze, SQL, databázový model, vymáhání, odpis, insolvence, zesplatnění úvěru

Design of relational database for managing written-off customers

Abstract

The methodology for theoretical part of this bachelor thesis will be based on the study and analysis of technical sources and experience from praxis. Based on the knowledge gathered in theoretical part of this thesis, in practical part will be identified optimal steps and structure of relational database. Results of this thesis will be based on the synthesis of results from practical and theoretical part.

Theoretical part of this thesis is focused on general problematic of write-offs and their origin and the methodology of designing a relational database.

The goal that this thesis sets for practical part is to create a design of relational database for managing of written-off files and customers for the potential implementation into standardized workflow of claim-companies and non-banking institutions.

Keywords: relational database, SQL, database model, recoveries, write-off, insolvency, repayment of the loan

Obsah

1. Úvod

Počet zadlužených osob, které nejsou schopny splácet své závazky v České republice, vzhledem k ekonomické situaci od začátku období výskytu viru COVID-19, roste. S tím souvisí i množství osob v exekucích, jejichž počet dle Exekutorské komory České republiky dosahoval v lednu roku 2024 přibližně 645 000 jedinců (Exekutorská komora ČR, 2024). Finanční instituce tak musí čelit nejen zvyšující se potencionální ztrátě zisků, ale i celkovému úbytku klientů, kteří nejsou schopni splácet své závazky a o jejichž správu se v konečném případě musí postarat stát.

Proto potřebují bankovní, nebankovní i vymáhající společnosti neustále zdokonalovat nejen procesy výběrů a vymáhání, ale též systémy pro správu svých klientů. Cílem je, aby co nejefektivněji získaly co možná nejvyšší část ušlého zisku zpět či klienty rozplatily před potencionálním úpadkem.

K těmto účelům se používají CRM[1] aplikace založené na data-driven[2] přístupech, ve kterých je možné nejen sledovat historii a vývoj dluhu klientů, ale též testovat či aplikovat různé vymáhací strategie a analyzovat efektivitu výběrů za pomoci reportingových nástrojů.

Cílem této práce je vytvoření návrhu databáze pro nejmenovanou nebankovní společnost, která přechází na novější CRM systém pro správu klientů v po-odpisovém vymáhání. Databáze bude zároveň vytvořena v jazyce SQL v programu SQL Server management studio™ od společnosti Microsoft. Funkčnost databáze bude otestována na vložených datech.

Jelikož se napříč trhem používají shodné technologie či postupy, může tato práce sloužit jako vzor pro nebankovní a vymáhající společnosti, které se chystají modernizovat své procesy a systémy.

[1] Customer relationship management, tedy aplikace pro správu klientů a vztahů s nimi (Chlebovský, 2005)

[2] Metoda rozhodování založená na sběru a analýze dat. (Zedníček, 2024)

2. Cíl práce a metodika

2.1. Cíl Práce

Cílem této práce je návrh relační databáze pro správu odepsaných klientů za použití SQL Server Management Studia™ od společnosti Microsoft. Tento model bude sloužit nejen účelům vybrané společnost, ale též jako možný vzor pro vymáhací a nebankovní instituce.

Mezi vedlejší cíle práce poté patří popsání problematiky vymáhacích procesů a návrhu relační databáze, společně s vytvořením testovacího prostředí a ověřením funkčnosti vytvořeného modelu na testovacích datech.

2.2. Metodika

Teoretická část této práce je založena na studiu odborných informačních zdrojů a praktických zkušenostech v daném pracovním prostředí a s prací v SQL Server Management Studiu™. Na základě syntézy teoretických poznatků, praktických zkušeností a výsledků praktické časti budou formulovány závěry práce.

Součástí praktické části je také ověření funkčnosti databáze na testovacích datech.

3. Teoretická východiska

Společnost, pro kterou je databáze připravována, se zabývá vymáháním odepsaných klientů. Ty získává přenosem odepsaného klienta z mateřské nebankovní společnosti, která fyzické osobě úvěr poskytla. Zjednodušený cyklus správy převzatých pohledávek u zmíněné firmy vypadá následovně:

1) Klient je odepsán v externí společnosti
2) Klientská data jsou převedena do databáze vymáhací firmy
3) Je vytvořena konsolidovaná složka aktuálně převzatých pohledávek klienta
4) Je připravena série strategií pro vymáhání na danou odpisovou složku
5) Probíhá vymáhání po určitou dobu (např. 3 roky)
6) Vymáhání je ukončeno a klientská složka je:
    a. Úspěšně uzavřena / Dluh je ukončen
    b. Otevřena ale o správu vymáhání se stará jiná instituce (Insolvenční správce)
    c. Neúspěšně ukončena a přeprodána

Tento cyklus se neustále opakuje a s plynutím času se může stát, že jeden klient má v dané společnosti například 3 různé odpisové složky s rozdílným počtem a výší pohledávek.

Aby bylo možné řádně a zcela připravit databázi pro účely této společnosti, je zapotřebí vycházet z teoretických základů a požadavků zmíněné firmy, které jsou rozebrány v následujících oddílech. První část se věnuje stručné teorii a rozebírání pojmů z prostředí odpisů, které budou využity k definici základních tabulek databáze.

Druhá část této práce pojednává o obecné teorii relační databáze, ze které bude čerpáno pro vytvoření vztahů a logiky mezi daty a jednotlivými tabulkami.

3.1. Pozadí vymáhacích procesů

Situace produktů na finančním trhu v České republice je velice komplikovaná a není obsahem této práce zmínit veškeré nuance úvěrových produktů, jejich řešení a definic. Abychom však mohli řádně nadefinovat databázi pro správu odepsaných klientů, je zapotřebí nejprve specifikovat základní pojmy, definice a logické okolnosti převzetí takovýchto klientů do vymáhacího procesu. (Veselá et al., 2021).

Nejtypičtějším produktem, se kterým se setkáme v rámci správy klientů po odpise je takzvaný spotřebitelský úvěr pro fyzické osoby. Ten ČNB popisuje následující definicí:

„Spotřebitelským úvěrem se rozumí odložená platba, peněžitá zápůjčka, úvěr nebo obdobná finanční služba poskytovaná nebo zprostředkovaná spotřebiteli.“ (Česká národní banka)

Tyto produkty poskytuje většina nebankovních finančních institucí za účelem zisku. Procesy vymáhání se zaměřují na řešení fyzických osob, které si pořídily zmíněný finanční produkt a nebyly schopny dodržet smluvní podmínky.

3.1.1. Výběry a Vymáhání (Collections and Recoveries)

Každá společnost má jiné strategie a procesy v rámci výběrů a vymáhání, které nejčastěji vycházejí z vlastní interní analýzy, ale zároveň dodržují doporučení České národní banky a institucí pomáhajících spotřebiteli (např. Člověk v tísni). Pro orientaci na obecném trhu je dobré uvést následující definice.

Výběry: Do doby, kdy je fyzická osoba aktivním klientem v portfoliu finanční instituce a předpokládá se u ní plnohodnotné dodržení stanovených závazků či splátek v rámci daného finančního produktu, veškeré finanční toky, které přichází ze strany klienta, řadíme do sekce *„Výběrů“*, neboli *„Collections“*. Lze proto říci, že do výběrů řadíme veškeré částky na splátku úvěru, které si instituce od klienta zařídí sama. A to i v případě že klient provádí řádnou splátku sám od sebe, je vyzván upomínkovým e-mailem, nebo je kontaktován prostřednictvím klientského centra o vynechané splátce.

Vymáhání: Pokud standardní výběry neuspěly a správu nad doplacením klientova závazku přebírá třetí strana, ať už určené oddělení, dceřiná, anebo soukromá společnost, hovoříme o takzvaném procesu *„Vymáhání“*, neboli *„Recoveries“*. Tyto třetí strany mají většinou agresivnější strategie na získání finančních prostředků od klienta, neboť se snaží vyzískat co nejvíce finančních prostředků před samotným finančním úpadkem klienta, po kterém správu nad závazky povětšinou přebírá stát.

V našem případě je třetí stranou dceřiná společnost zaměřená na vymáhání odepsaných klientů.

3.1.2. Odpisy

Z pohledu práce s převzatými pohledávkami chápeme odpisy dle definice České bankovní asociace jako: *„Rozhodnutí věřitele o odepsání úvěru z bilance v situaci, kdy je půjčka nesplatitelná. Děje se tak prostřednictvím redukce aktiv o hodnotu půjčky.“* (Česká bankovní asociace, 2024).

Úvěr klienta je vyřazen z aktivního portfolia, zároveň je ukončena historie jeho úvěru v databázi a půjčka, či skupina produktů, bývá označena za odepsanou („WO“ nebo též „Write-off“). Následně je možné tuto pohledávku předat třetím stranám, ať již prodejem, nebo interní předávkou.

Odepsané úvěry nesou detailní informace o zůstatku úvěru, poplatcích navyšujících hodnotu zůstatku a dalších detailech, které budou zmíněny v praktické části. Je běžnou praxí, že v případě odpisu se uzavírají všechny produkty klienta v dané společnosti, i když ostatní produkty mohl splácet řádně a delikventní byl pouze u jedné.

Právní, účetní a procesní kroky předcházející samotnému odpisu jsou mimo rozsah této práce.

3.1.3. Odpisová složka

Odpisová složka je souborem produktů (úvěrů) daného klienta, které byly převzaty z mateřské společnosti v jeden čas. Pokud měl například klient dvě půjčky u mateřské společnosti, je vytvořen jeden soubor s unikátním identifikátorem, který obsahuje detaily obou úvěrů.

V průběhu vymáhání je následně pracováno právě s odpisovou složkou, na kterou jsou uplatňovány veškeré strategie a výpočty spjaté s vymáháním. Zároveň slouží jako zdroj informací pro kontrolní úřady.

Ve vymáhacím cyklu jednoho klienta se může vyskytovat více odpisových složek. To může nastat v případě, že bylo vymáhání klienta řádně ukončeno a následně si vzal u mateřské společnosti úvěr nový.

V praktické části bude odpisová složka jádrem tvorby databáze, neboť veškeré procesy budou vycházet z údajů na odpisové složce.

3.1.4. Insolvenční rejstřík

Pro správu odepsaných klientů je důležitým zdrojem informací také tzv. Insolvenční rejstřík (ISIR).

Jedná se o veřejně přístupný seznam subjektů, které procházejí insolvenčním

řízením. Jsou zde dle insolvenčního zákon č. 182/2006 Sb. vedeny spisy subjektů s jednotlivými zápisy procesu. Detaily jsou uchovávány po dobu 5 let po ukončení insolvenčního řízení. K prvnímu zápisu dojde ve chvíli, kdy sám dlužník podává návrh na zahájení insolvenčního řízení. O schválení či zamítnutí návrhu rozhoduje příslušný krajský soud (info@aion.cz).

Během vymáhacího procesu proto může dojít k tomu, že si klient podá návrh na insolvenční řízení. Jestliže je krajským soudem schválen, je dlužníkovi přidělen insolvenční správce, který se stará o vymáhání dluhu a přerozdělování financí věřitelům. Na začátku schválení insolvenčního řízení se ovšem musí přihlásit všichni věřitelé, kteří od dlužníka požadují splacení jeho závazku v rámci insolvenčního řízení a ukončit vlastní procesy vymáhání. Věřitelé musejí dodat patřičné podklady a insolvenční správce pak rozhoduje o tom, zdali má věřitel na základě dodaných podkladů právo se k vymáhání přihlásit a v jakém rozsahu (Veselá et al., 2021).

Z výše uvedených důvodů bude v praktické části uvedena i tabulka pro insolvenční rejstřík, do které se budou ukládat data čerpaná z databáze ISIR, aby bylo možné identifikovat klienty, kteří si zažádali o insolvenční řízení. Tím pádem nepřijít o možnost získání finančních prostředků zpět.

3.1.5. CRM Software

Customer Relationship Management (CRM) v překladu znamená „správa (řízení) vztahů se zákazníkem“. Jedná se o interaktivní proces, během kterého se společnost snaží stanovit rovnováhu mezi vlastní investicí a uspokojením potřeb zákazníka. Optimum této rovnováhy se stanovuje maximálním ziskem obou stran (Chlebovský, 2005, s. 23).

Aplikace, které umožňují výše zmíněný proces realizovat, se nazývají CRM aplikace, nebo CRM software. Tyto programy shromažďují veškeré potřebné informace o klientech, jejich finančních tocích, kontaktních údajích apod. Zároveň umožňují řídit komunikaci s klientem (e-maily, telefonáty, SMS, atd.) při snaze zachovat s ním dobré vztahy, pokud je to možné.

Vymáhací společnost, pro kterou je připravována tato databáze, má CRM software vytvořený na míru externí společností. Mezi hlavní požadované funkce společnost zařadila:

a) Zobrazování potřebných dat během hovoru s klientem (zobrazení odpisové složky s veškerými osobními údaji a finančními toky)
b) Automatické zasílání zpráv klientovi přes 3 kontaktní zdroje (e-mail, SMS,

dopis)

c) Řízení strategií, vymáhacích kroků a automatického vytáčení hovorů
d) Zasílání informací o dohodě s klientem vymáhacím pracovníkům
e) Správu dohod uzavřených s klientem

3.1.6. Shrnutí požadavků

Pro vytvoření vhodné databáze pro tyto účely musíme vycházet z výše uvedených informací a požadavků a nadefinovat tabulky, které budou uchovávat následující informace:

a) Odpisové složky
b) Transakce a finanční stavy
c) Dohody s klientem
d) Smluvní informace
e) Osobní a kontaktní údaje klienta
f) Vymáhací strategie
g) ISIR data
h) Informace o zaměstnancích
i) Historie úkonů ve vymáhání

Detailněji bude o návrhu databáze pojednáno v praktické části.

3.2. Teorie databází

Tato teoretická část je zaměřena převážně na práci s relačními databázemi. Pro zasazení do širšího kontextu je zde však uvedena i obecná teorie databází a databázových modelů.

3.2.1. Přístupy ke zpracování dat

Při práci s databázemi operujeme s daty, která můžeme chápat jako jednotlivé údaje zaznamenané v systému a určené ke zpracování. Z historického hlediska se během vývoje databázových systémů postupně vyvíjely různé pohledy na samotné zpracování hromadných dat. Tyto tendence se lišily zejména v přístupu, jakým bude s daty nakládáno (zdali budou data propojena mezi sebou, jaké prvky budou tvořit jednotný celek apod.). Musely být také zohledňovány výpočetní zdroje dostupné v dané době a následně i konvence v informačních technologiích.

Z pohledu míry vzájemné integrovanosti organizace dat a stupně nezávislosti programů na způsobu jejich uložení rozlišuje Vostrovský (2014, s. 5) následující vývojové skupiny:

a) Konvenční přístupy:
    i. Agendové zpracování dat
    ii. Integrované zpracování dat
b) Databázové přístupy:
    i. Databázové zpracování dat
    ii. Objektové zpracování dat

3.2.1.1. Agendové zpracování

Agendové neboli souborové zpracování je jedním z prvních přístupů k práci s daty a databázemi. Tato technologie započala v 70. letech 19. století. Na soubory bylo nahlíženo jako na jeden individuální celek tak, jak je tomu v operačních systémech. Každý soubor obsahoval jednotky informací, zvané záznamy, které nesly informace o jedné entitě. V souboru se vyskytovala data jednoho typu entit a měly přímý uživatelský přístup. (Merunka, Vostrovský, 1998, s. 7)

Data se zpracovávala za pomoci samostatných izolovaných úloh malého formátu, které se nazývají *agendy*. Tento způsob ovšem nebyl velmi efektivní, jelikož každá agenda pracovala s daty individuálně a nedocházelo k žádnému propojování (integrovanosti) dat či agend. To vedlo ke generování velkého množství souborů a zbytečnému kopírování dat. Při potřebě data upravit pak docházelo ke komplikacím a zpomalování průběhu zpracování (Vostrovský, 2014, s. 6).

3.2.1.2. Integrované zpracování

Další vývojovou fází ve zpracování dat bylo takzvané integrované zpracování. Aby se snížila datová redundance[3], začaly nově vytvářené agendy využívat datové soubory agend už vytvořených. Tím se zvýšila míra integrovanosti dat. Zpracovatelské programy však zůstaly nadále nezávislé, jak tomu bylo v agendovém zpracování (Vostrovský, 2014, s. 10).

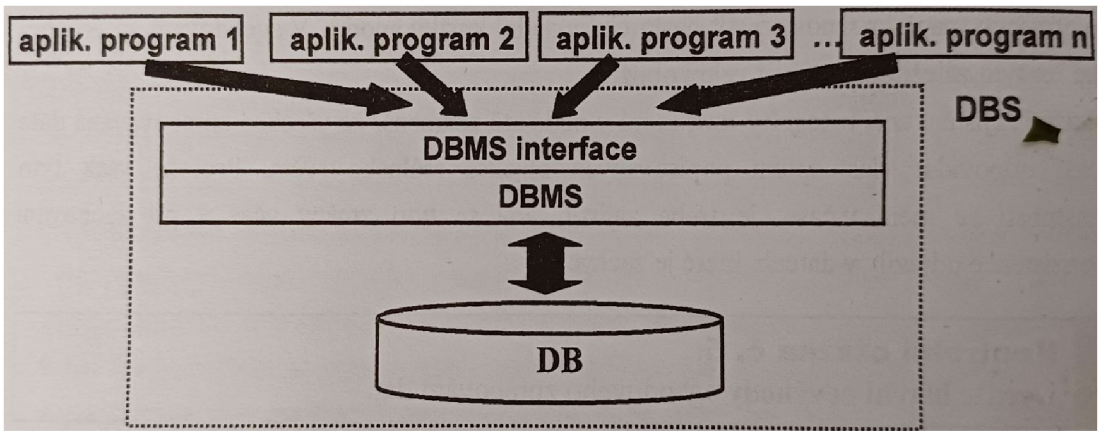
3.2.1.3. Databázové zpracování

S nástupem velkorozměrový počítačů v 70. letech se zvýšila výpočetní kapacita a efektivita zpracování dat. Využívání agend se tak ukázalo jako neefektivní a bylo nahrazeno databázovým zpracováním. To pracuje s logicky uspořádanými a ucelenými bázemi dat, ke kterým mají přístup programy v dané společnosti. Tyto programy pracují

[3] Nadbytečné vícenásobné ukládání dat (pozn. autora)

s jedním datovým zdrojem a zpracování dat je tak konzistentní a rychlejší než v předchozích případech. Dle nastavení práv mají navíc uživatelé přístup jen k datům, která jim jsou užitečná. Data jsou tak nejen chráněna od vnějšího přístupu, ale více uživatelů může užít jeden datový zdroj zároveň (Vostrovský, 2014, s. 11).

Vizuální zobrazení principu databázového zpracování:



Obrázek 1- Databázové zpracování, zdroj: Vostrovský, 2014, s. 10

V praktické části je užit právě výše jmenovaný databázový přístup. Ačkoli se nejedná o nejmodernější technologii k řízení a zpracování dat, je na Českém trhu hojně využívána. Zároveň je to technologie společnosti, pro kterou je v praktické části databáze připravována.

3.2.1.4. Objektové zpracování

Nejmodernějším pohledem na zpracování dat je takzvané objektové zpracování. Na informace je v tomto případě nahlíženo jako na objekty v systému. Báze dat mohou být přímo integrovány do objektově orientovaného programovacího jazyka (OOPJ), který s daty manipuluje. Pro zpracování informací proto není potřeba data dotazovat přes dílčí kroky, kterými může být například jazyk SQL. Datový model objektově orientované databáze nese aspekty OOPJ jako třídy, atributy a integritní omezení. Báze dat aplikace se shoduje s datovým modelem, což vede k efektivnější udržitelnosti výsledného kódu (Friedrichsen et al. 2020).

3.2.2. Databázové systémy

Databázové systémy (DBS) představují v dnešní době základ pro programové vybavení osobních počítačů. Podřazeným pojmem databázových systémů jsou samotné databáze (DB), které lze definovat jako soubory dat k určité problematice. Uvnitř databází jsou data dále organizována. Ve většině případů navíc spolu data logicky souvisí. Tyto

informace jsou uchovávány a využívání k různým účelům, například vytváření reportů, kontaktování klientů, analýze prosperity firmy apod. Díky své struktuře a příslušným programům umožňují, narozdíl od původních kartoték, rychlé zpracování, úpravu a uchovávání velkého množství informací. Aby mohlo být s daty v DB operováno, je však zapotřebí také systém řízení báze dat (SŘBD). Tento program má na starosti popis a manipulaci s daty v bázi. Pro zjednodušení je možné říci, že je databázový systém nadřazenou množinou databáze a systému řízení báze dat. Jako komunikační rozhraní mezi daty a uživatelem se využívají programovací (např. Python) nebo dotazovací (např. SQL) jazyky (Vostrovský, 2014).

3.2.3. Systém řízení báze dat

Jak bylo zmíněno, SŘBD je software, skrz který uživatelé komunikují s uloženými daty. Dle Friedrichsenové (2020) patří mezi hlavní funkce řádného systému řízení báze dat:

a) Efektivní zpracování velkého množství dat
b) Řízení a provádění dotazů
c) Synchronizace a aktualizace dat

U větších společností není neobvyklé využití více než jednoho SŘBD uvnitř firemní struktury. Nebo dokonce jednoho systému, ale v různých verzích. Příkladem může být mateřská společnost zadavatele, jejíž datový sklad je na bázi Oracle uložen v Polsku a data se odtud stahují do České republiky na SQL Server™ od společnosti Microsoft. Datoví analytici se tak musí orientovat v obou systémech, kvůli zachování čistoty a pravdivosti dat. V praktické části bude použit jmenovaný SQL Server™ od společnosti Microsoft.

Následující přehled ukazuje nejpopulárnější relační systémy řízení báze dat.:

Tabulka 1- Populární relační SŘBD, zdroj: Friedrichsen et al, 2020, s. 11

Name	Company	Website
Oracle	Oracle Corporation	oracle.com/database/
MySQL	Oracle/Open source	mysql.com/
SQL Server	Microsoft	microsoft.com/en-us/sql-server/default.aspx
PostgreSQL	Open source	postgresql.org/
Db2	IBM	ibm.com/products/db2-database
SQLite	Open source	sqlite.org
MariaDB	Open source	mariadb.org/
Access	Microsoft	products.office.com/en-us/access

3.2.4. Relační databáze

Výběr datového modelu ovlivňuje dotazovací jazyk a konstrukci dotazů do databáze. Tato práce pojednává o tvorbě databáze relační. Proto se zaměříme pouze na její definování. Tradiční definici relačních databází nabízí Pokorný (1992, s. 72): *„Databáze založená na relačním modelu, v němž jsou data logicky uspořádána do relací, tj. výsledků kartézského součinu nad doménami neboli množinami údajů.“.*

Fyzickou reprezentací uvedené relace je dvourozměrný objekt tvořený řádky a sloupci (např. tabulka). Na tabulce budou v následující podkapitole ilustrovány hlavní struktury relačních databází pro lepší pochopení zmíněné definice.

3.2.4.1. Struktury v relačních databázích

Mezi důležité pojmy v relačních databází řadíme:

a) Doména
b) Entita
c) Atribut
d) Kandidátní klíč
e) Primární klíč (PK)
f) Cizí klíč (FK)
g) Kompozitní klíč
h) Relace
i) Vztahy mezi entitami
j) Index
k) Relační model

Pro vizuální reprezentaci nadcházejících pojmů v relačních databázích využijeme tabulku s fiktivními daty, viz *tabulka 2*. V příkladové tabulce s názvem Clients budeme sledovat záznamy o klientech.

Tabulka 2 - Vzorová tabulka Clients, zdroj: vlastní zpracování

ClientID	Clientname	Street	City	State	Zip	Government
20240001	David Vostrý	Testová 116	Testra	ČR	9999 99	NULL
20240002	Tomáš Havelka	Testová 12	Testra	ČR	1000 99	NULL
20240003	Lucie Šedá	Příkladová 14	Testra	ČR	1001 99	NULL

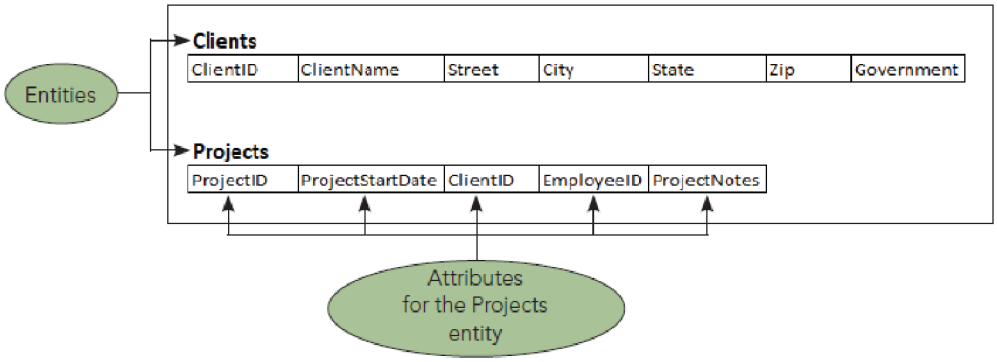
Doména: Doména je souborem všech přístupných dat a informací, které mají stejný významový typ (Vostrovský, 2014, s.34). Což můžeme chápat jako množinu dostupných hodnot pro daný sloupec, jakými jsou název, typ a rozsah.

Relace: Jedná se o databázovou tabulku (nebo jakoukoli jinou strukturu s řádky

a sloupci, (Friedrichsen et al, 2020)), která vyjadřuje podmnožinou karteziánského součinu nad množinami domén a vztahů mezi prvky těchto domén (Vostrovský, 2014, s.34).

Entita a atribut: Tabulka i sloupce jsou v databázi pro uživatele identifikovány svými jmény. Jednotlivé řádky tabulky obsahují soubory údajů o takzvaných *entitách*, tedy sledovaných objektech (klientech). Sloupce označujeme jako *atributy* neboli záznamy stejného typu (Pokorný, 1992).

Klíče: K identifikaci konkrétního unikátního řádku tabulky nám slouží *kandidátní klíče*. Může se jednat o jeden či více sloupců, tedy jeden či více kandidátních klíčů. Tyto nemusí vycházet pouze z jednoho sloupce, ale mohou být kombinací více atributů (*kompozitní klíč*). Tento postup však není doporučován, jelikož ztěžuje dotazování. Z kandidátních klíčů je vybrán jeden (na základě bezpečnosti, funkcionality, apod.), jakožto *Primární klíč*. V našem případě můžeme za primární klíč považovat údaje ve sloupci ClientID. Funkčnost ostatních polí by měla být závislá na primárním klíči. Vztah s jinou tabulkou v databázi je možné nadefinovat prostřednictvím takzvaného *cizího klíče*. Pokud by se tedy v jiné tabulce v databázi (viz entita Projects na obrázku 2) nacházel také údaj ClientID, ale nedefinoval zde unikátní řádek (ClientID by bylo v dané tabulce duplicitní), hovořili bychom v dotyčné tabulce o ClientID jako o cizím klíči (Friedrichsen et al, 2020).



Obrázek 2 - Vztahy mezi entitami, zdroj: Friedrichsen et al,2020, s.4

Vztahy mezi entitami: Vtahy mezi entitami jsou definovány za pomoci vztahu mezi primárním a cizím klíčem. Toto spojení je vyjádřeno matematicko-logickým zobecněním jako počet množin z jedné tabulky vůči počtu množin v jiné tabulce, které tvoří vzájemný propojení. Kupříkladu vztah mezi tabulkami Clients a Projects na obrázku 2. Klient je zde napojen na všechny své projekty v tabulce Projects a množina některých

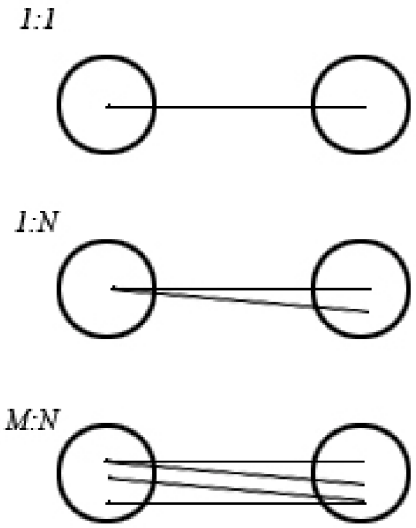
projektů z tabulky Projects je napojena k jednomu klientovi z tabulky Clients. Z pohledu tabulky Clients, kde je ClientID primárním klíčem, vůči tabulce Projects tak hovoříme o vztahu jeden k mnoha (Friedrichsen et al, 2020).

Variace těchto vztahů budou podrobněji rozebrány v kapitole 3.2.5. Kardinalita a parcialita.

Indexy: Další důležitou datovou strukturou je *index*. Indexy umožňují rychlý přístup k datům díky hodnotám, které jsou k nim během indexace přiřazeny. Tyto hodnoty tvoří *klíč indexu* a jsou (narozdíl od klíčů samotných, které jsou součástí logiky databáze) uloženy na disku. Způsob fungování indexů je velmi podobný knižním rejstříkům, jelikož program zobrazuje pouze data ze zvoleného indexu. Tím dochází k rychlejšímu zpracování. Indexy si vytváří sám uživatel. Nevýhodou indexů je jejich zabírání kapacity na disku a nutnost manuálního aktualizování při změnách struktury tabulky či dat (Soukup; Krásenský, 1998, s. 225-227).

Relační model: Z výše uvedených definic a pojmů je možné zjednodušeně shrnout pojem relačního modelu. Relační model je, v daném kontextu, sdružením dat do relací, které tvoří základ relační databáze. Sdružení tabulek (relací), funkčních vztahů mezi nimi, definovaných indexů, klíčů a ostatních komponent pak tvoří relační databázi.

3.2.5. Kardinalita a parcialita



Obrázek 3 - Kardinalita, zdroj: vlastní zpracování

Hovoříme-li o vztahu mezi tabulkami v databázi, považujeme *kardinalitu* za vztahy mezi dvěma nebo více entitami. Tyto vztahy mohou nabývat třech podob, a to jeden-k-jedné (1:1), jeden-k-mnoha (1:N), mnoho-k-mnoha (M:N). Prvním záznamem

je definována kardinalita u nadřazené (rodičovské) tabulky s primárním klíčem. Druhý záznam vyjadřuje kardinalitu u podřazené (dětké) tabulky s klíčem cizím (Friedrichsen, et al, 2020, s. 217). Grafické vyjádření kardinality je na obrázku č. 3.

Vztah jeden-k-jedné udává, že k jednomu záznamu z jedné tabulky náleží pouze jeden záznam z tabulky druhé. Příkladem vztahu 1:1 může být případ omezených bankovních účtů. U dané banky si může klient otevřít pouze jeden účet. Čímž by mezi entitou klientů a entitou účtů existoval vztah 1:1.

Kardinalita jeden-k-mnoha vyjadřuje, že k jednomu záznamu z jedné tabulky náleží *n* množství záznamů z tabulky druhé. Vztah 1:N představuje například vztah mezi klientem a úvěry. Jeden klient může mít *n* množství úvěrů, ale daný úvěr smí být napsán jen na jednoho klienta.

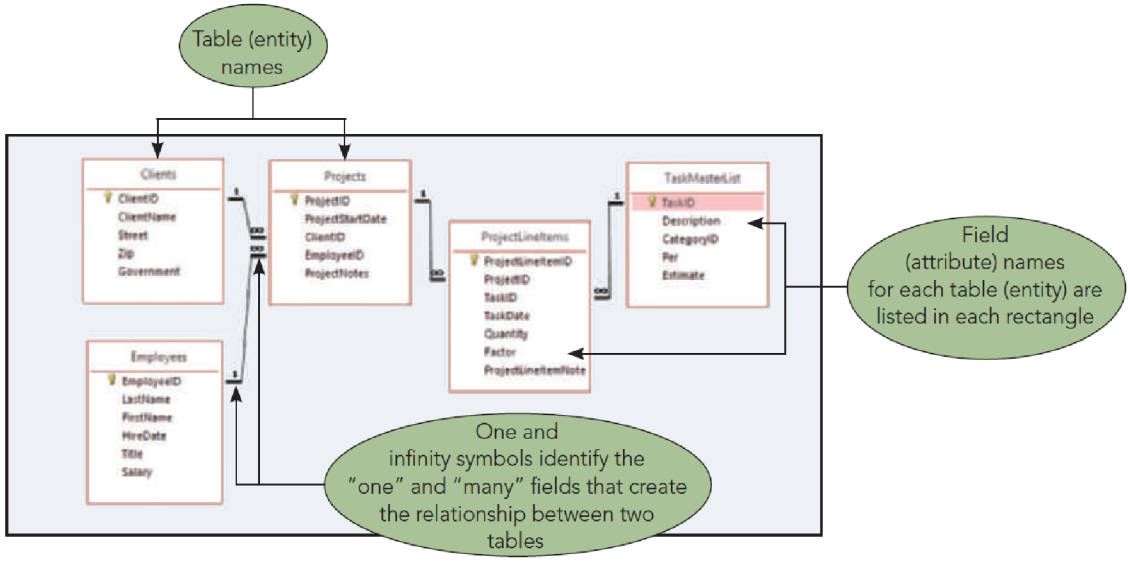
Spojení mnoho-k-mnoho říká, že k *m* množství záznamů jedné tabulky lze přiřadit *n* množství záznamů z tabulky druhé. Kardinalitu M:N pak můžeme vyjádřit kupříkladu vztahem mezi klienty a bankovními poradci. Více klientů může být spojeno s více bankovními poradci, pokud se ti střídají. A více bankovních poradců může *n* množství klientů. Kardinalita M:N nemůže být v databázi přímo vytvořena, i když tuto vazbu mají prakticky všechny databáze. K implementaci vztahu mnoho-k-mnoha v databázi musí být mezi dvěma entitami vložena třetí, takzvaná kompozitní entita. Přímo v databázi je kompozitní entita nazývána jako propojovací tabulka a k jejímu vytvoření je třeba primárních klíčů z obou propojovaných tabulek (Friedrichsen et al, 2020, s. 217-218). V našem příkladu by tedy musela být vytvořena třetí tabulka, např. SpravaKlientu, kde by v jednom sloupci byly identifikační čísla klientů a v druhém identifikační čísla poradců.

Parcialita je termín přímo spojený s kardinalitou, neboť udává, jestli jsou určené vztahy povinné či nikoli. Můžeme proto říci, že kardinalita určuje maximální počet vztahů a parcialita počet minimální.

3.2.6. ER Diagram a ER Model

Způsobů znázornění ER diagramů a ER modelů je od různých autorů více. Mezi známé konvenční techniky patří například Bachmanova, Barkerova, Martinova,UML notace a pod.. V rámci zachování integrity tohoto textu jsou využity definice od Friedrichsenové a kolektivu (2020).

ER Diagram - ER Diagram (ERD) neboli entity-relationship[4] diagram, je grafické znázornění reprezentující vztahy mezi entitami databáze. Tato vizualizace napomáhá nejen k vytváření, ale i analýze samotného návrhu databáze. Čtverce v ERD reprezentují entity s atributy. Linie mezi spojenými entitami ukazují vztahy mezi nimi za pomoci číselných údajů jako 1 a ∞, viz obrázek č. 4 (Friedrichsen et al, 2020).



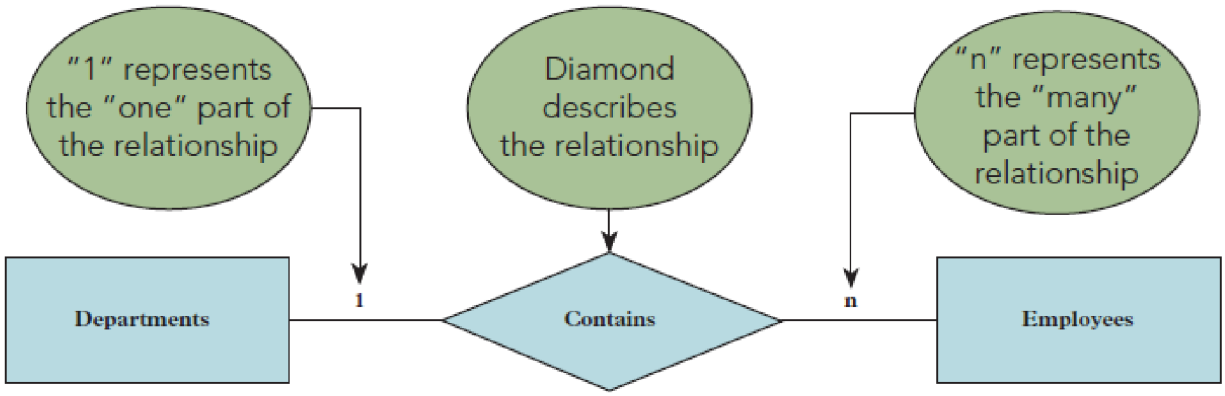
Obrázek 4 - ER Diagram, zdroj: Friedrichsen, 2020, s. 11

Kardinalita mezi atributy je vytvářena pomocí společných polí. Tato pole se nemusí ve spojených tabulkách jmenovat stejně, avšak shodné pojmenování klíčových polí je osvědčeným postupem při vytváření databází. (Friedrichsen et al, 2020, s. 214).

Pojící pole z jedné tabulky na straně kardinality „jeden" ve vztahu jeden-z-mnoha je vždy primárním klíčem dané tabulky. Na straně „mnoho" je dané pole definováno jako cizí klíč. Cizí klíč nesmí být nikdy zároveň primárním klíčem na straně „jeden" v dotyčné tabulce (Friedrichsen et al, 2020, s. 214).

ER Model – ER modely (ERM) neboli entity-relationship modely jsou další možností dokumentování vztahů relační databáze. Ve svém designu využívají obdélníky pro definování entit a diamanty (nebo kosočtverce) pro určení vztahu mezi entitami. Linie pojící tyto objekty nesou informaci o typu kardinality mezi entitami (Friedrichsen et al, 2020, s. 215). Základní varianta ERM je znázorněna na obrázku č. 5.

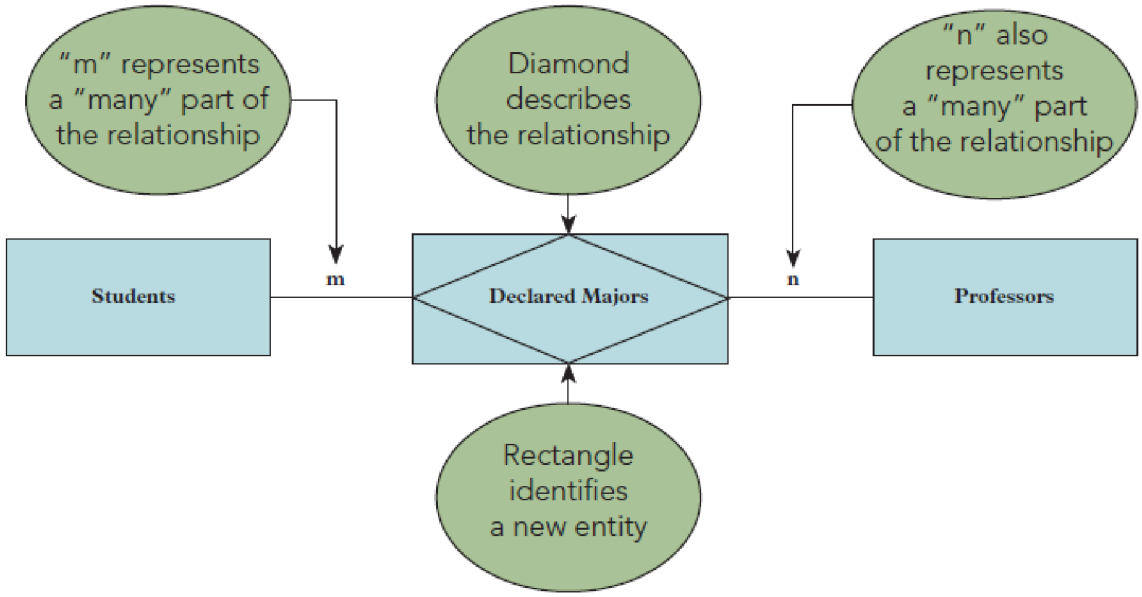
[4] vztah mezi entitami



Obrázek 5 - ERM pro vztah jedna-k-jedné, zdroj: Friedrichsen et al, 2020, s. 218

Pokud bychom chtěli v ERM ukázat kardinalitu M:N, je zapotřebí vytvořit kompozitní entitu sloučenou z obdélníku a diamantu. Takto je následně reprezentováno fyzické vytvoření pojící tabulky, viz obrázek č. 6.

Atributy v ERM mohou být znázorněny jako ovály připojené liniemi k entitám. Každý atribut je v separátním oválu. Primární klíče se mezi atributy rozlišují podtržením názvu daného atributu (Friedrichsen et al, 2020, s. 218).



Obrázek 6 - ERM mnoho-ku-mnoha, zdroj: Friedrichsen et al, 2020, str. 218

3.2.7. Coddova pravidla

Tato pravidla publikoval Edgar F. Codd na počátku 70. let. Přesto jsou platná dodnes. Pravidel je celkem 12 a slouží k efektivnímu fungování relačních modelů. (Vostrovský, 2014, s.21)

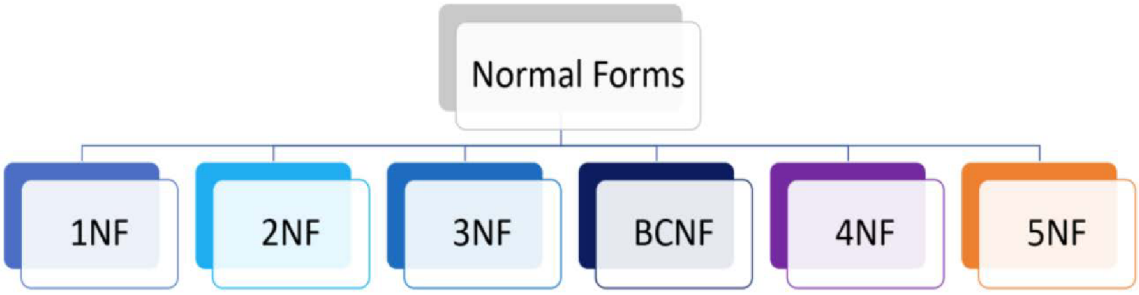
Seznam těchto pravidel klasifikuje Vostrovský (2014, s. 21-22) následovně:

1) Pravidlo SŘDB – Databázový systém musí spravovat data pouze relačními operacemi
2) Pravidlo informační – Data musejí být reprezentována na logické úrovni pouze jako hodnoty v relačních tabulkách
3) Pravidlo zajišťující přístup – Každý údaj musí být dosažitelný pomocí kombinace názvu tabulky, názvu sloupce a hodnoty primárního klíče.
4) Pravidlo zpracovatelnosti neznámých hodnot – Každá neznámá hodnota musí být dosažitelná jinými známými hodnotami.
5) Pravidlo relačního katalogu – Popis databáze musí být reprezentován relačním způsobem, jako tabulka
6) Pravidlo pro jazyk – SŘDB musí mít minimálně jeden počítačový jazyk umožňující definici dat, integritní omezení, manipulaci s daty, prací s transakcemi a pravidla přístupu.
7) Pravidlo pohledů – musí být umožněny pohledy do databáze a aktualizace dat
8) Pravidlo operací – všechny relační operace musejí pracovat s celými tabulkami
9) Pravidlo fyzické a logické nezávislosti – změny ve struktuře tabulek nesmí ovlivnit výsledky operací.
10) Pravidlo nezávislosti dat na integritních omezeních – Změny v integritních omezeních nesmí ovlivnit výsledky operací.
11) Pravidlo nezávislosti dat na distribuci – výsledky operací nesmějí být ovlivněny konkrétním rozmístěním dat
12) Pravidlo nenarušitelnosti SŘDB – rozhraní SŘDB nesmí být narušeno aplikací, ani uživatelem.

Codd pro definování databázových systémů používá matematickou koncepci relační algebry. Díky té rozděluje data na jednotlivé množiny a podmnožiny, které spolu vzájemně souvisí (Stephens et al, 2012, s. 33).

3.2.8. Normalizace databáze

Normalizace je postupný proces, při kterém se odstraňují redundantní a opakovaná data v databázi. Databáze mohou obsahovat data, která se nacházejí ve více tabulkách najednou. To by mohlo vést k bezpečnostním incidentům, zpomalení dotazů, ohrožení kapacity úložiště, zhoršení efektivity aktualizací a narušení integrity dat (Stephens et al, 2012, s .231).



Obrázek 7 - Normální formy, zdroj: guides.visual-paradigm.com

3.2.8.1. Normální formy

Normální formy (NF) reprezentují úroveň, do jaké byla databáze normalizována. Proces vychází z nenormalizované databáze. V prvních třech formách každý následný stupeň normalizace následuje až po krocích v normální formě předchozí. Pokud je tedy databáze normalizována do třetí úrovně, musely proběhnout normalizační kroky druhé a předtím i prvním úrovně (Stephens et al, 2012, s .231).

Obrázek 7 znázorňuje většinu známých normálních forem, mimo formu nenormalizovanou, nultou. Stephens (Stephens et al, 2012) a Friedrichsenová (Friedrichsen et al, 2020, s. 194) ve svých publikacích uvádí, že v relačních databázích jsou typicky užívány první tři normální formy, ačkoli oficiální zdroje rozeznávají úrovní více. Proto budou pouze tyto tři formy podrobněji popsány v následujících odstavcích. Pro úplnost však bylo zapotřebí vyšší úrovně normalizace alespoň zmínit, i když jsou mimo rámec této práce.

3.2.8.1.1. 1. Normální forma

Tabulka je považována za normalizovanou v 1.NF, když neobsahuje žádné opakující se hodnoty v atributech, každý sloupec obsahuje atomické hodnoty[5] a v tabulce

[5] Prvky, které není možné rozdělit na menší části

nejsou žádné duplicity (Friedrichsen et al, 2020, s. 171). Toho dosáhneme logickým rozřazením dat a určením primárních klíčů v tabulkách (Stephens et al, 2012, s. 233).

3.2.8.1.2. 2. Normální forma

Relaci považujeme za normalizovanou ve druhé NF, pokud se již nachází v první NF a neklíčové atributy jsou plně závislé na celém poli primárního klíče. Druhá NF tedy organizuje sloupce do menších relací (Friedrichsen et al, 2020, s. 186). Toho dosáhneme uložením dat, která jsou jen částečně závislá na primárním klíči, do separátní tabulky (Stephens et al, 2012, s. 233).

3.2.8.1.3. 3. Normální forma

Tabulka ve třetí NF musí zahrnovat normalizaci ve druhé NF. Zároveň se v ní nesmí objevovat tranzitivní závislosti. Za tranzitivní závislost považujeme stav, kdy jeden neklíčový atribut determinuje jiný neklíčový atribut (Friedrichsen et al, 2020, s. 190). Cílem této formy je odstranění dat, která nejsou závislá na primárním klíči (Stephens et al, 2012, s. 233).

3.2.8.2. Denormalizace databáze

Normalizace může vést ke snížení celkového výkonu databáze. Denormalizace je procesem, při kterém snižujeme úroveň normalizace databáze o jednu až dvě úrovně za účelem zpětného navýšení výkonu. Nevýhodou tohoto procesu je redundance dat a extrémní náročnost realizace referenční integrity (Stephens et al, 2012, s. 238).

3.2.9. SQL

Jazyk SQL (Structured Query Language – strukturovaný dotazovací jazyk) je standardním jazykem pro datovou manipulaci a získávání informací z relační databáze. Na rozdíl od programovacího jazyka jsou jeho funkce omezené a zaměřené na práci databázemi. Svojí strukturou je blízký principu kladení otázek v anglickém jazyce. Kromě příkazů na zobrazování dat nabízí možnosti vytváření nových databází, tabulek, polí a indexů. Dále umožňuje mazat a vkládat záznamy. Standart ANSI SQL poskytuje navíc funkce pro manipulaci s daty. (Stephens et al, 2012).

Dle Vostrovského (2014, s. 55) obsahuje standard jazyka SQL tři hlavní komponenty:

1) DDL (Data definition language) - Jazyk pro definici dat, který umožňuje skládat příkazy pro definici nových dat a jejich popisu.

2) DML (Data manipulation language) – Jazyk pro manipulaci dat umožňující dotazování nad daty v databázi, vkládání dat nových a rušení a změny dat existujících. Příklad příkazů DML jsou SELECT, INSERT, UPDATE, ALTER

3) DCL (Data control language) – Jazyk pro řízení dat, díky kterému je možné přiřazovat práva uživatelům a řízení transakcí. Příkladem příkazů DCL jsou GRANT a REVOKE.

Čtvrtou komponentu přidává Stephens et al (2010):

4) TCL (Transaction control language) – jazyk pro zpracování transakcí umožňuje kontrolu a řízení transakcí v databázi. Jako příklady TCL jsou uvedeny příkazy COMMIT a ROLLBACK.

Typický příklad DML příkazu jazyka SQL uvádí Mollinaro (2009, s.28):

```
SELECT *
FROM Emp
WHERE deptno = 10
```

Lingvistické vyjádření: „Zobraz všechny zaměstnance (záznamy) z tabulky Emp, kteří pracují v oddělení (deptno) 10.“

Jazyk SQL má řadu implementací s rozdílnými vlastnostmi. Mezi nejpopulárnější implementace řadíme MySQL, Oracle, Microsoft SQL Server™, Sybase a IMB DB2 (Stephens et al, 2012).

V praktické části je využit program SQL Server™ od společnosti Microsoft, který pracuje s variací jazyka SQL zvaným Transact-SQL (T-SQL). Tento jazyk má specifický dialekt a umožňuje využívat databázové nástroje od společnosti Microsoft (Hotek, 2009).

Detailní popis použitého jazyka SQL není v rozsahu této práce. V přílohách je však uveden celý kód použitý k definování databáze, primárních klíčů, nahrání testovacích dat a zkušební skript.

4. Praktická část práce

V kapitole 3.1.6. Shrnutí požadavků, byly vypsány požadované vlastnosti relační databáze pro vymáhací společnost. V následujících kapitolách bude představen ER diagram logického návrhu databáze, popis jednotlivých tabulek a test funkčnosti připraveného SQL skriptu. Vždy se vychází z požadavků společnosti. Databáze byla pojmenována WO_Management.

4.1.Popis problematiky

Databáze pro společnost vymáhající odepsané složky bude dle zadavatele centralizována kolem tabulky obsahující údaje o odpisových složkách. Původní historická verze byla centralizována kolem historické tabulky obsahující všechny informace o odpisové složce a registrovaných smlouvách. Pro získání potřebné informace pro analytiky a fungování původního softwaru tak muselo docházet k sebe-napojování (self-join) zmíněné tabulky a definování řady podmínek, pro získání jedné specifické informace. Toto nastavení vedlo nejen k redundanci dat a pomalejšímu zpracovávání původní aplikací, ale též zatěžování databáze komplikovanými skripty. Návrh vytvořený v této práci bude dle požadavků více normalizovaný za účelem řádného chodu nové CRM aplikace a zjednodušení práce s databází pro datové analytiky.

4.2.ER Diagram

Na obrázku č. 8 je vytvořen detailní ER diagram navrhované databáze. Větší zobrazení ER diagramu je v příloze č. 1. Jednotlivé relace mají uvedeny své názvy, atributy, datový typ atributu a omezující podmínky mezi které řadíme primární klíče (uvedené jako PK), cizí klíče (uvedené jako FK), kandidátní klíče (zkratka AK), hodnoty generované databází (AI) a indexy (zkratka IX). Označení *NULL* v návrhu říká, že daný atribut může obsahovat prázdnou hodnotu. Tedy že nemusí být v řádku vyplněn. Vztahy mezi relacemi jsou naznačeny přerušovanými liniemi, které vedou od tabulky s primárním klíčem (rodičovská tabulka) k tabulce s cizím klíčem (dětská tabulka). Dětská tabulka je na konci linie určena tečkou. Každá linie obsahuje název vztahu mezi tabulkami. Tento název reprezentuje spojení mezi tabulkami, tedy primární a cizí klíč ve spojených tabulkách. Stejný název je následně využit ve skriptu v příloze k definování vztahu. Kardinalitu a parcialitu relací udávají hodnoty uvedené na konci přerušovaných linií u tečky.

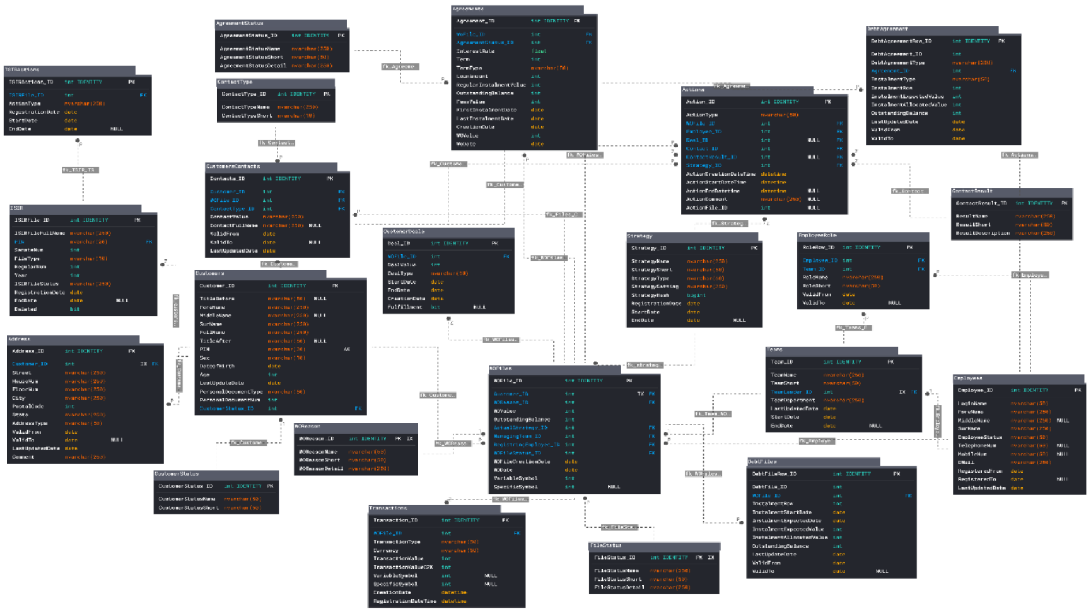
P = vztah jeden ku jeden a více

Z = vztah jeden ku nula a více

1 = vztah jedna ku jedné

0-1 = vztah jeden ku nula či jedné

Na obrázku č. 8 je možné vidět, že je návrh pomyslně centralizován kolem tabulky *WOFiles*, která obsahuje přijaté odpisové složky. Další tabulkou je tabulka *Customers*, která uchovává osobní údaje klienta. Na tuto je napojen seznam adres klienta, záznamy z insolvenčního rejstříku a registrované kontakty na klienta v rámci dané odpisové složky. Další sekcí napojenou na tabulku *WOFiles,* jsou jednotlivé smlouvy obsažené v odpisových složkách. Tabulka se nazývá *Agreements* a jsou na ni napojeny záznamy o splácení na dané smlouvě. Další sekcí jsou akce a strategie. Tabulka *Actions* reprezentuje všechny aktivity na odpisovou složku včetně záznamů o hovorech, odeslaných e-mailech apod. Tabulka je napojena na strategie a informace o zaměstnancích. Poslední sekcí jsou *DebtFiles* (informace o historii dluhu na složce), *CustomerDeals* (individuální dohody s klientem) a *Transactions* (transakce provedené v rámci odpisové složky). Návrh obsahuje celkem 22 tabulek. Z toho 6 tabulek obsahuje pouze doplňující informace k atributu. Tyto nazýváme číselníky a jsou vytvořeny na žádost zadavatele. Detailnější popis relací je uveden v následující sekci.



Obrázek 8 - ER Diagram databáze WO_Management, zdroj: vlastní zpracování

4.3. Tabulky relační databáze

Specifické definice relací jsou popsány v tabulkách u příslušné relace. *Column name* značí název atributu. *Type* udává datový typ sloupce. *Properties* značí databázový typ atributu (viz 4.2. ER Diagram). Ve sloupci *Description* je detailnější popis významu atributu. Tento popis je ve skriptu SQL v příloze zadán i přímo do databáze, aby mohl uživatel sloupce identifikovat. *UI* v popisu značí Unikátní Identifikátor. Společnost požaduje, aby byly textové řetězce ve formátu NVARCHAR (variabilní formát čísel a textu) a většina číselných hodnot a klíčů ve formátu INT (numerická hodnota integer). Hodnoty obsahující pouze ukazatele 1 nebo 0 jsou označovány formátem BIT. Datum ve formátu *„den-měsíc-rok“* mají datový typ DATE. Pokud je k hodnotě formátu DATE přidán navíc čas, jedná se o datový typ DATETIME. Čísla v závorkách za názvy datových typů udávají maximální povolenou délku řetězce hodnoty.

4.3.1. WOFiles

Pomyslné jádro zpracovávané databáze tvoří tabulka *WOFiles* (viz tabulka č.3), na kterou se bude odkazovat CRM systém, jako na první. Primárním klíčem tabulky je *WOFile_ID*, které reprezentuje unikátní identifikátor odpisové složky. Dále je zapotřebí uvést unikátní klíč klienta, důvod odpisu složky a celkovou hodnotu odpisové složky při přijetí.

Tabulka 3 - Tabulka WOFiles, zdroj: vlastní zpracování

WOFiles			
Column name	**Type**	**Properties**	**Description**
WOFile_ID	int	PK, AI	UI Odpisové složky
Customer_ID	int	FK	UI Klienta
WOReason_ID	int	FK	UI Důvodu odpisu složky
WOValue	int		Celková hodnota odepsané částky na všech smlouvách
OutstandingBalance	int		Zbývající částka k výběru
ActualStrategy_ID	int	FK	UI Aktuálně užité strategie
ManagingTeam_ID	int	FK	UI týmu, který má složku ve správě
RegistringEmployee_ID	int	FK	UI registrujícího zaměstnance
WOFileStatus_ID	int	FK	UI Stavu odpisové složky
WOFileCreationDate	date		Datum vytvoření odpisové složky
WODate	date		Datum odpisu všech smluv v mateřské společnosti
VariableSymbol	int		Variabilní symbol složky pro transakce
SpecificSymbol	int	NULL	Specifický symbol odpisové složky

OutstandingBalance reprezentuje aktuální dluh na složce. Též je zapotřebí uvést aktuální vymáhací strategii, pod kterou je složka vedena. Tato strategie definuje například

způsob komunikace s klientem během hovoru na oddělení centrálního vymáhání pohledávek, textaci odeslaných e-mailů, apod. *ManagingTeam_ID* značí aktuální tým, který má složku na starosti. Složka během své existence může přejít například z vymáhání na oddělení insolvencí. Tabulka musí dle požadavků obsahovat také identifikátor zaměstnance, který složku založil, aktuální stav odpisové složky (otevřená, zavřená, apod.), datum založení složky a datum dokončení odpisu/ů u mateřské společnosti. Variabilní a specifický symbol slouží k účetní alokaci (přiřazování) transakcí k dané odpisové složce. Je třeba uvést fakt, že jeden klient může mít ve vymáhání více odpisových složek najednou.

Složky v tabulce *WOFiles* se tvoří během přijímání balíčku odepsaných smluv z mateřské společnosti.

4.3.2. DebtFiles

Tabulka č. 4 ukazuje detail zmíněné relace. Pokud je klient přijat se svými smlouvami na vymáhání, je vytvořen nový předpis splátek pro danou odpisovou složku. Dluhy jsou tak konsolidovány (sloučeny) do jednoho a je proto třebavytvořit nový předpis splátek. Tabulka *DebtFiles* obsahuje informace o předepsaných splátkách na danou odpisovou složku a informace o jejich umořování. *DebtFileRow_ID* zde tvoří primární klíč identifikující unikátní řádek záznamů. *DebtFile_ID* je unikátní identifikátor rozpisu dluhu.

Tabulka 4 - Tabulka DebtFiles, zdroj: vlastní zpracování

DebtFiles			
Column name	**Type**	**Properties**	**Description**
DebtFileRow_ID	int	PK, AI	UI řádku Rozpisů
DebtFile_ID	int		UI složky rozpisu dluhu
WOFile_ID	int	FK	UI Odpisové složky
InstalmentRow	int		Pořadí předepsané splátky dluhu
InstalmentStartDate	date		Datum začátku očekávání splátky
InstalmentExpectedDate	date		Poslední datum očekávané splátky
InstalmentExpectedValue	int		Hodnota očekávané splátky
InstalmentAllocatedValue	int		Celková přiřazená částka ke splátce
OutstandingBalance	int		Zbývající nedoplatek
LastUpdateDate	date		Datum aktualizace dat řádku
ValidFrom	date		Datum začátku platnosti předpisu
ValidTo	date	NULL	Datum konce platnosti předpisu

Jelikož tabulka obsahuje vypsané splátky, kterých je více než jedna na odpisovou složku i rozpis dluhu, bude *DebtFile_ID* duplicitní. Relace obsahuje informace o odpisové složce, které je předpis přidělen, pořadí splátky, očekávané datum splátky a očekávané částce splátky. Dále jsou zde údaje o alokované částce na splátku získané z účetního

softwaru a aktuální dluh na konci splátkového období dané splátky. Tabulka musí obsahovat i informaci o poslední aktualizaci dat na daném řádku a data dne začátku a konce platnosti předpisu.

4.3.3. Transactions

Tabulka *Transactions* (viz tabulka č. 5), představuje všechny transakce na odpisovou složku, které byly registrovány do systémy přes účetní software. Jedná se o transakce přijaté (splátky klienta) i odeslané, či přičtené na složku (vrácení přeplatku, připsání úroku z prodlení). Primárním klíčem transakcí je *Transaction_ID* – unikátní identifikátor jedné transakce. Dále je identifikována přiřazená odpisová složka. Transakce musí uvádět také typ dané transakce (splátka, poplatek apod.), měnu (klientům je smluvně umožněno posílat splátku v cizí měně), celkovou částku transakce a částku transakce v měně CZK (česká koruna). Příchozí transakce mají dále přiřazen variabilní a specifický symbol, ale u odchozích transakcí jej není třeba uvádět. *CreationDateTime* nás informuje o datu a čase, kdy byla transakce odeslána. *RegistrationDateTime* specifikuje datum a čas zanesení transakce do systému. Během víkendů mohou například nastat prodlevy v čase odeslání transakce a jejího přijetí do společnosti.

Tabulka 5 - Tabulka Transactions, zdroj: vlastní zpracování

Transactions			
Column name	**Type**	**Properties**	**Description**
Transaction_ID	int	PK, AI	UI Provedené transakce
WOFile_ID	int	FK	UI Odpisové složky
TransactionType	nvarchar(50)		Typ transakce dodávaný účetním softwarem
Currency	nvarchar(50)		Měna provedené transakce
TransactionValue	int		Číselná hodnota transakce
TransactionValueCZK	int		Číselná hodnota transakce v Kč
VariableSymbol	int	NULL	Variabilní symbol transakce
SpecificSymbol	int	NULL	Specifický symbol transakce
CreationDateTime	datetime		Datum a čas odeslání transakce
RegistrationDateTime	datetime		Datum a čas zanesení transakce do systému

4.3.4. Customers

Relace (viz tabulka č. 6) obsahující osobní údaje registrované na konkrétního klienta. Primárním klíčem je *Customer_ID,* unikátní identifikátor klienta. Kandidátním klíčem je zde *PIN*, jedná se o rodné číslo klienta. Tento údaj musí být uveden jako textový řetězec, jinak by došlo k ořezání hodnot u klientů narozených po roce 2000. *PIN* je důležitý zejména pro práci s tabulkou ISIR. Pro lepší zpracovávání textací zpráv klientovi obsahuje tabulka separátně titul před jménem, křestní jméno, prostřední jméno, příjmení a titul

za jménem. Sloupec *FullName* skládá jméno do jedné buňky kvůli požadavku na zobrazování v CRM systému. Dalšími údaji jsou datum narození a věk klienta, typ a název registrovaného dokladu a stav klienta ve společnosti. Stav klienta je definován slovně skrz číselníkovou tabulku. Relace obsahuje i datum poslední změny údajů na řádku.

Tabulka 6 - Tabulka Customers, zdroj: vlastní zpracování

Customers			
Column name	**Type**	**Properties**	**Description**
Customer_ID	int	PK, AI	UI Klienta
TitleBefore	nvarchar(50)		Titul před jménem
ForeName	nvarchar(250)		Křestní jméno klienta
MiddleName	nvarchar(250)	NULL	Prostřední jméno klienta
SurName	nvarchar(250)		Příjmení klienta
FullName	nvarchar(250)		Plné jméno klienta
TitleAfter	nvarchar(50)		Titul za jménem klienta
PIN	nvarchar(20)	AK	Rodné číslo klienta
Sex	nvarchar(10)		Pohlaví klienta
DateofBirth	date		Datum narození klienta
Age	int		Věk klienta
LastUpdateDate	date		Datum poslední aktualizace údajů
PersonalDocumentType	nvarchar(50)		Typ registrovaného osobního dokladu
PersonalDocumentNum	int		Číslo osobního dokladu
CustomerStatus_ID	int	FK	ID aktuálního stavu klienta

4.3.5. CustomerContacts

Relace (viz tabulka č. 7) obsahuje seznam kontaktních úrovní na klienta na odpisovou složku (mimo adresy). Redundance údaje o odpisové složce je v požadavku zadavatelské společnosti, protože na každou odpisovou složku může mít jeden klient registrovány jiné kontaktní údaje. Primárním klíčem tabulky je *Contacts_ID*. Tabulka musí obsahovat UI klienta, ke kterému je kontakt připojen. *ContactType_ID* se odkazuje na číselník a udává typ kontaktní informace (e-mail, telefon, telefon do zaměstnání, telefon na manželku apod.). *ContactValue* je znění daného kontaktu (např. vyplněný e-mail nebo konkrétní telefonní číslo). *ContactFullName* udává, pokud je třeba, jméno pro oslovení kontaktu.

Dalšími informacemi jsou datum vložení kontaktu do systému, datum skončení platnosti kontaktu a poslední datum aktualizace daného řádku. Informace slouží k určení kontaktů, které je aktuálně možné u daného případu využít.

Tabulka 7 - Tabulka CustomerContacts, zdroj: vlastní zpracování

CustomersContacts			
Column name	**Type**	**Properties**	**Description**
Contacts_ID	int	PK, AI	UI kontaktu na klienta
Customer_ID	int	FK	UI klienta
WOFile_ID	int	FK	UI Vymáhací složky
ContactType_ID	int	FK	UI typu kontaktu
ContactValue	nvarchar(250)		Uložená hodnota kontaktu na klienta
ContactFullName	nvarchar(250)	NULL	Jméno kontaktní osoby
ValidFrom	date		Datum uložení kontaktu
ValidTo	date	NULL	Datum ukončení platnosti kontaktu
LastUpdateDate	date		Datum poslední aktualizace řádku

4.3.6. Address

Tabulka č. 8 ukazuje historizující tabulku adres na klienta. Během vymáhacího procesu může klient měnit své kontaktní i trvalé adresy. Z tohoto důvodu je struktura kontaktních i trvalých adres vystavěna v separátní tabulce s údaji o datech platnosti dané adresy. Primárním klíčem tabulky je unikátní identifikátor záznamu adresy *Address_ID*. Pro identifikaci trvalé či kontaktní adresy je uveden parametr *AddressType*. Mimo základních adresních údajů, je v tabulce uvedeno také datum platnosti dané adresy od a do. Předposledním atributem je *LastUpdatedDate*, tedy datum poslední změny údajů řádku. Do sloupce *Comment* mohou klienti zadávat detaily k dané smlouvě (např. „Zeleně označený zvonek.“). Délka komentáře je tak omezena nativně na 4000 znaků (Hotek, 2009). Údaje o adresách jsou modifikovatelné operátory přes CRM systém.

Tabulka 8 - Tabulka Address, zdroj: vlastní zpracování

Address			
Column name	**Type**	**Properties**	**Description**
Address_ID	int	PK, AI	UI adresy
Customer_ID	int	FK	UI klienta
Street	nvarchar(250)		Název ulice
HouseNum	nvarchar(250)		Číslo popisné
FloorNum	nvarchar(250)		Patro/podlaží
City	nvarchar(250)		Vesnice/Město
PostalCode	int		Poštovní směrovací Číslo
State	nvarchar(250)		Stát
AddressType	nvarchar(50)		Typ adresy (trvalá, kontaktní)
ValidFrom	date		Datum registrace adresy
ValidTo	date	NULL	Datum ukončení platnosti adresy
LastUpdatedDate	date		Datum poslední aktualizace řádku
Comment	Nvarchar(250)		Komentář k adrese

4.3.7. ISIR

Relace (viz tabulka č. 9) obsahuje data, denně stahovaná z insolvenčního rejstříku. Primárním klíčem relace je UI složky insolvenčního rejstříku, *ISIRFile_ID*. *ISIRFullName* je celý název složky ISIRu a je tvořen sloučením sloupců s identifikátory z rejstříku. Těmi je číslo senátu, typ složky, regulérní číslo spisu a rok založení spisu. Pro napárování tabulky na příslušného klienta je zapotřebí UI klienta. K zjištění aktuálního stavu spisové složky slouží atribut *ISIRFileStatus*. Abychom mohli zjistit datum přihlášení klienta k insolvenci, je přidán atribut *RegistrationDate*. Konec insolvenčního řízení *EndDate* je definován jako datum s možností prázdné hodnoty, než konec řízení nastane. *Deleted* je booleanová (0 nebo 1) hodnota říkající, zda byl již záznam smazán z ISIRu. Datový typ je proto *bit*.

Tabulka 9 - Tabulka ISIR, zdroj: vlastní zpracování

ISIR			
Column name	**Type**	**Properties**	**Description**
ISIRFile_ID	int	PK, AI	Unikátní identifikátor ISIR složky
ISIRFileFullName	nvarchar(250)		Celý název ISIR složky složený z SenatenNum+Filetype+RegularNum+/+Year
PIN	nvarchar(20)	FK	Rodné číslo registrované osoby
SenateNum	int		Číslo senátu
FileType	nvarchar(10)		Typ složky, nativně "ISIR"
RegularNum	int		Regulerní číslo ISIR složky
Year	int		Rok založení ISIR spisu
ISIRFileStatus	nvarchar(250)		Aktuální stav ISIR složky
RegistrationDate	date		Datum registrace insolvence
EndDate	date	NULL	Konec insolvenční řízení
Deleted	bit		Smazáno z ISIR; 1/0

4.3.8. ISIRActions

Tabulka 10 - Tabulka ISIRActions, zdroj: vlastní zpracování

ISIRActions			
Column name	**Type**	**Properties**	**Description**
ISIRActions_ID	int	PK, AI	UI Akce ISIR
ISIRFile_ID	int	FK	Unikátní identifikátor ISIR složky
ActionType	nvarchar(250)		Typ akce ISIR řízení
RegistrationDate	date		Datum registrace aktivity
StartDate	date		Začátek data rozhodnutí
EndDate	date	NULL	Konec data rozhodnutí

V tabulce *ISIRActions* (tabulka č. 10) jsou obsažena data na denní bázi stahována z ISIRu. Data obsahují informace o krocích provedených v rámci insolvenčního řízení. Těmi může být schválení nebo zamítnutí insolvence apod. Primárním klíčem

je *ISIRActions_ID*, unikátní identifikátor provedené akce na složce. Cizím klíčem je *ISIRFile_ID* vztažený k tabulce *ISIR*. Typ akce ISIR řízení říká, jaký byl provedený úkon. Datum registrace je datum uvedení aktivity do systému. Atribut *StartDate* je datem začátku platnosti rozhodnutí a *EndDate* vyjadřuje datum konce aktivity. Některé aktivity datum konce nemusí obsahovat, proto je vlastností atributu *EndDate* NULL.

4.3.9. Actions

Tabulka 11 - Tabulka Actions, zdroj: vlastní zpracování

Actions			
Column name	**Type**	**Properties**	**Description**
Action_ID	int	PK, AI	UI Aktivity
ActionType	nvarchar(50)		Typ Akce na klienta
WOFile_ID	int	FK	UI Odpisové složky
Employee_ID	int	FK	UI Spravujícího zaměstnance
Deal_ID	int	FK, NULL	UI provedené Dohody s klientem
Contact_ID	int	FK	UI kontaktu na klienta
ContactResult_ID	int	FK, NULL	ID Výsledku aktivity
Strategy_ID	int	FK	UI užité strategie
ActionCreationDateTime	datetime		Datum a čas vytvoření aktivity
ActionStartDateTime	datetime		Začátek prováděné aktivity
ActionEndDatetime	datetime	NULL	Konec prováděné aktivity
ActionComment	nvarchar(250)	NULL	Komentář k provedené akci
ActionFile_ID	int	NULL	UI souboru uloženého na souborovém serveru

Relace *Actions* (viz tabulka č. 11) obsahuje všechny aktivity učiněné v rámci vymáhání na klienta u dané odpisové složky. Těmito aktivitami mohou být odeslání e-mailu, telefonický hovor, odeslaný dopis apod. Primárním klíčem tabulky je *Action_ID,* unikátní identifikátor provedené akce. Typ provedené aktivity je zadán přes CRM software. Cizími klíči jsou informace o odpisové složce, identifikátor akcí provádějícího zaměstnance, UI případné dohody s klientem, ID použitého kontaktu, ID výsledku aktivity, identifikátor přiloženého souboru (např. pdf soubor textu e-mailu) odkazujícího na složku v CRM systému a ID strategie, ve které se v tu dobu složka nacházela. Dalšími informacemi jsou datum a čas vytvoření aktivity, datum a čas začátku aktivity, datum a čas konce aktivity. Poslední položkou jsou případné dodatečné komentáře od zaměstnance.

4.3.10. CustomerDeals

Tabulka č. 12 obsahuje detail dohod uzavřených s klientem na dané odpisové složce během provedené aktivity. Dohoda s klientem může být například snížení splátek na určité období nebo příslib platby v dohodnuté částce. Primárním klíčem je UI vytvořené

dohody, *Deal_ID*. Relace obsahuje dále odkaz na odpisovou složku, dohodnutou částku a typ dohody. Důležitými atributy jsou *StartDate* a *EndDate*, které specifikují časový rozsah uzavřené dohody. Předposledním atributem je datum vytvoření dohody požadované zadavatelem. Ačkoli by bylo možné získat datum vytvoření dohody z tabulky *Actions,* analytici vyžadují pro separátní reportu *CreationDate* v tabulce *CustomerDeals*. Jedná se o formu zadavatelem požadované denormalizace databáze. Atribut *Fulfillment* je booleánským parametrem vyjadřujícím, jestli byla dohoda naplněna.

Tabulka 12 - Tabulka CustomerDeals, zdroj: vlastní zpracování

CustomerDeals			
Column name	**Type**	**Properties**	**Description**
Deal_ID	int	PK, AI	UI dohody s klientem
WOFile_ID	int	FK	UI Odpisové složky
DealValue	int		Dohodnutá částka s klientem
DealType	nvarchar(50)		Typ dohody s klientem (PTP, Odklad)
StartDate	date		Datum začátku dohody s klientem
EndDate	date		Datum konce dohody s klientem
CreationDate	date		Datum vytvoření dohody
Fulfillment	bit	NULL	Naplnění dohody

4.3.11. Strategy

Relace *Strategy* (tabulka č. 13) obsahuje informace o specifickém nastavení strategií užívaných při vymáhání. Tato data čerpá CRM systém pro správu automatizovaných aktivit. Data také slouží například operátorům pro stanovení postupu vymáhání.

Tabulka 13 - Tabulka Strategy, zdroj: vlastní zpracování

Strategy			
Column name	**Type**	**Properties**	**Description**
Strategy_ID	int	PK, AI	UI strategie
StrategyName	nvarchar(250)		Celý název strategie
StrategyShort	nvarchar(50)		Název strategie zkr.
StrategyType	nvarchar(50)		Typ strategie
StrategySetting	Nvarchar(250)		Nastavení pravidel strategie vyjádřeno textem
StrategyHash	bigint		Hash kód strategie s nastavením pro účely CRM softwaru
RegistrationDate	date		Datum registrace strategie do systému
StartDate	date		Datum začátku platnosti strategie
EndDate	date	NULL	Datum konce platnosti strategie

Strategy_ID je primární klíč tabulky. Dalšími atributy jsou název strategie, zkratka názvu strategie, typ strategie (Soft, Medium, Hard), textový řetězec obsahující nastavení strategie a číselná kombinace *StrategyHash,* která kódem definuje nastavení strategie

pro CRM software.

Tabulka musí obsahovat také datum uvedení strategie do systému a datum začátku a konce užití strategie v systému.

4.3.12. Agreements

V této relaci (tabulka č. 14) jsou obsaženy informace o konkrétních smlouvách na odpisové složce. Primárním klíčem je unikátní identifikátor smlouvy, *Agreement_ID*. Cizími klíči jsou ID odpisové složky a informace o stavu smlouvy (doplacená, otevřená). Dalšími informacemi o smlouvě jsou úroková sazba, počet jednotek doby trvání smlouvy (číselná hodnota), typ jednotek trvání smlouvy (týdny, měsíce), celková výše půjčky, výše řádné splátky, zbývající nedoplatek, hodnota poplatků v době odpisu, datum první předpokládané splátky, datum poslední smluvní splátky a datum podpisu smlouvy. Dále je třeba uvést i celkovou hodnotu odpisu na dané smlouvě a datum odpisu smlouvy ze systému mateřské společnosti. Tabulka slouží pro dokládání jednotlivých dluhů klientovi či kontrolním orgánům.

Tabulka 14 - Tabulka Agreements, zdroj: vlastní zpracování

Agreements			
Column name	**Type**	**Properties**	**Description**
Agreement_ID	int	PK, AI	UI čísla smlouvy
WoFile_ID	int	FK	UI přiřazené odpisové složky
AgreementStatus_ID	int	FK	UI stavu klienta
InterestRate	float		Úroková sazba smlouvy
Term	int		Počet jednotek doby trvání smlouvy
TermType	nvarchar(50)		Typ jednotek trvání smlouvy dodávaný mateřskou spol. - měsíční/týdenní
LoanAmount	int		Celková zapůjčená částka
RegularInstalmentValue	int		Částka pravidelných splátek
OutstandingBalance	int		Zbývající částka k doplacení
FeesValue	int		Celková částka předepsaných poplatků v době odpisu
FirstInstalmentDate	date		Datum první splátky na smlouvě
LastInstalmentDate	date		Datum poslední splátky na smlouvě
CreationDate	date		Datum podepsání smlouvy
WOValue	int		Celková odpisová hodnota smlouvy. Součet OutstandingBalance a FeesValue
WoDate	date		Datum odpisu smlouvy z mateřské společnosti

4.3.13. DebtAgreement

Jedná se o detail řádných splátek a dluhů na smlouvě. Tabulka (viz. tabulka č. 15) obsahuje nejen historii splácení smlouvy před odpisem, ale též předepsané splátky

a alokace plateb po odpise. Primárním klíčem je UI řádku tabulky *DebtAgreementRow_ID*. Atribut *DebtAgreement_ID* odkazuje na ID souboru předpisu v CRM systému. Sloupec *DebtAgreementType* identifikuje typ předpisu (předodpisový, poodpisový apod.). Jako cizí klíč slouží *Agreement_ID*. Dalšími informacemi jsou typ předepsané splátky, pořadí předepsané splátky, očekávaná hodnota splátky, alokovaná částka a zbývající dluh na smlouvě v době konce předepsané splátky. Tyto informace dodává účetní software. Datumové atributy tabulky zaznamenávají datum začátku a konce období předepsané splátky a poslední datum aktualizace řádku.

Tabulka 15 - Tabulka DebtAgreement, zdroj: vlastní zpracování

DebtAgreement			
Column name	**Type**	**Properties**	**Description**
DebtAgreementRow_ID	int	PK, AI	UI řádku rozpisů dluhů na smlouvě
DebtAgreement_ID	int		UI Odkazující na PDF předpis v daném typu předpisu v CRM softwaru
DebtAgreementType	Nvarchar(250)		Typ předpisu - předodpisový, poodpisový
Agreement_ID	int	FK	UI smlouvy
InstalmentType	nvarchar(50)		Typ předepsané splátky dodávaný účetním systémem - Předodpisová/Poodpisová
InstalmentRow	int		Pořadí předepsané splátky
InstalmentExpectedValue	int		Očekávaná hodnota předepsané splátky
InstalmentAllocatedValue	int		Alokovaná částky plátky
OutstandingBalance	int		Zbývající dluh
LastUpdatedDate	date		Datum poslední aktualizace řádku
ValidFrom	date		Datum začátku platnosti předepsané splátky
ValidTo	date		Datum konce platnosti předepsané splátky

4.3.14. Employees

Tabulka 16 - Tabulka Employees, zdroj: vlastní zpracování

Employees			
Column name	**Type**	**Properties**	**Description**
Employee_ID	int	PK, AI	UI Zaměstnance
LoginName	nvarchar(50)		Přihlašovací jméno ZMT do CRM softwaru
ForeName	nvarchar(250)		Křestní jméno ZMT
MiddleName	nvarchar(250)	NULL	Prostřední jméno ZMT
SurName	nvarchar(250)		Příjmení Zaměstnance
EmployeeStatus	nvarchar(50)		Stav zaměstnance přidělený HR softwarem
TelephoneNum	nvarchar(50)	NULL	Telefonní číslo na ZMT do kanceláře
MobileNum	nvarchar(50)	NULL	Mobilní telefonní číslo zaměstnance
EMail	nvarchar(250)		E-mail na ZMT
RegisteredFrom	date		Datum registrace ZMT do systému
RegisteredTo	date	NULL	Datum konce registrace ZMT do systému
LastUpdatedDate	date		Datum poslední aktualizace údajů řádku

V této relaci (tabulka č. 16) jsou uloženy informace o zaměstnancích vymáhací společnosti. Primárním klíčem je *Employee_ID*.

Dalšími atributy jsou přihlašovací jméno do CRM systému, jméno, prostřední jméno, příjmení a stav zaměstnance. Jako kontaktní informace slouží pracovní e-mailová adresa, telefon do kanceláře a pracovní mobilní telefon. Atributy *RegisteredFrom* a *RegisterdTo* určují datum nástupu do zaměstnání a konce pracovního poměru. Také je uvedeno datum poslední aktualizace řádku. Tabulka slouží převážně k identifikaci zaměstnance v rámci prováděné aktivity.

4.3.15. EmployeeRole

Historické informace o pozici, kterou zastával zaměstnanec v určité době, jsou uložené v tabulce *EmployeeRole* (viz tabulka č. 17). Jelikož se jedná o historizující tabulku, primární klíčem je UI řádku tabulky. Jako cizí klíče jsou uvedeny ID zaměstnance a identifikátor týmu. Dále je uveden název pracovní pozice a zkratka daného názvu. Datumové hodnoty určují časový rámec, ve kterém zaměstnanec na dané pozici pracoval. Pokud je atribut *ValidTo* prázdnou hodnotou, znamená to, že zaměstnanec danou pozici stále zastává.

Tabulka 17 - Tabulka EmployeeRole, zdroj: vlastní zpracování

EmployeeRole			
Column name	**Type**	**Properties**	**Description**
RoleRow_ID	int	PK, AI	UI řádku pozice zaměstnance
Employee_ID	int	FK	UI Zaměstnance
Team_ID	int	FK	UI Aktuálního týmu zaměstnance
RoleName	nvarchar(250)		Název Pozice
RoleShort	nvarchar(50)		Zkratka názvu pozice
ValidFrom	date		Datum začátku platnosti přiřazené pozice ZMT
ValidTo	date	NULL	Datum konce platnosti přiřazené pozice ZMT

4.3.16. Teams

V tabulce č. 18 jsou uloženy informace o týmu, ke kterému je zaměstnanec přidělen. Primárním klíčem je atribut *Team_ID*. Cizím klíčem je *TeamLeader_ID*, který referuje na atribut *Employee_ID* v tabulce *Employee*. Dalšími atributy jsou název týmu, zkratka názvu týmu a oddělení, kterému tým náleží. Pro určení začátku a konce platnosti týmu jsou definovány atributy *StartDate* a *EndDate*. Posledním atributem je *LastUpdateDate* určující datum poslední aktualizace týmu.

Tabulka 18 - Tabulka Teams, zdroj:vlastní zpracování

Teams			
Column name	**Type**	**Properties**	**Description**
Team_ID	int	PK, AI	UI Týmu
TeamName	nvarchar(250)		Název týmu
TeamShort	nvarchar(50)		Zkratka názvu týmu
TeamLeader_ID	int	FK	UI vedoucího týmu
TeamDepartment	nvarchar(250)		Oddělení týmu
LastUpdatedDate	date		Datum poslední aktualizace řádku
StartDate	date		Datum založení týmu
EndDate	date	NULL	Datum ukončení týmu

4.4.Číselníky

Na žádost analytického oddělení, pro potřeby interního přidávání a úprav záznamů, byly vytvořeny takzvané číselníky. Jedná se o tabulky, jejich účelem je uživateli vysvětlit význam identifikátoru, nebo zobrazit pochopitelnější text, namísto číselného ID u daného atributu. CRM systém z nich čerpá při zobrazování informací uživateli. Detaily číselníků jsou uvedeny v tabulkách číslo 19,20,21,22,23 a 24.

Tabulka 19 - Číselník AgreementStatus, zdroj: vlastní zpracování

AgreementStatus			
Column name	**Type**	**Properties**	**Description**
AgreementStatus_ID	int	PK, AI	UI stavu smlouvy
AgreementStatusName	nvarchar(250)		Název stavu smlouvy
AgreementStatusShort	nvarchar(50)		Zkratka názvu stavu smlouvy
AgreementStatusDetail	nvarchar(250)		Detailní popis stavu smlouvy

Tabulka 20 - Číselník ContactType, zdroj: vlastní zpracování

ContactType			
Column name	**Type**	**Properties**	**Description**
ContactType_ID	int	PK, AI	UI Typu kontaktu
ContactTypeName	nvarchar(250)		Název typu kontaktu
ContactTypeShort	nvarchar(10)		Název typu kontaktu zkr.

Tabulka 21 - Číselník FileStatus, zdroj: vlastní zpracování

FileStatus			
Column name	**Type**	**Properties**	**Description**
FileStatus_ID	int	PK, AI	UI Stavu složky
FileStatusName	nvarchar(250)		Plný název stavu složky
FileStatusShort	nvarchar(50)		Zkrácený název stavu složky
FileStatusDetail	nvarchar(250)		Popis stavu složky

Tabulka 22 - Číselník WOReason, zdroj: vlastní zpracování

WOReason			
Column name	**Type**	**Properties**	**Description**
WOReason_ID	int	PK, AI	UI Důvodu odpisu
WOReasonName	nvarchar(50)		Název důvodu odpisu
WOReasonShort	nvarchar(50)		Název důvodu odpisu zkr.
WOReasonDetail	Nvarchar(250)		Detaily spojené s důvodem odpisu

Tabulka 23 - Číselník ContactResult, zdroj: vlastní zpracování

ContactResult			
Column name	**Type**	**Properties**	**Description**
ContactResult_ID	int	PK, AI	UI Výsledku kontaktu s klientem
ResultName	nvarchar(250)		Název Výsledku
ResultShort	nvarchar(50)		Název výsledku zkr.
ResultDescription	nvarchar(250)		Popis výsledku hovoru

Tabulka 24 - Číselník CustomerStatus, zdroj: vlastní zpracování

CustomerStatus			
Column name	**Type**	**Properties**	**Description**
CustomerStatus_ID	int	PK, AI	ID Stavu klienta
CustomerStatusName	nvarchar(50)		Název statusu klienta
CustomerStatusShort	nvarchar(50)		Název statusu klienta zkr.

4.5. Testování funkčnosti

Jelikož testování dat pro zobrazení v CRM aplikaci s úspěchem provedla firma poskytující tento software a množství dat v sytému nedosahuje ani poloviční zátěže procesních kapacit serverů, bude v rámci testování vytvořen report, pomocí kterého společnost ověřuje svoji výkonnost.

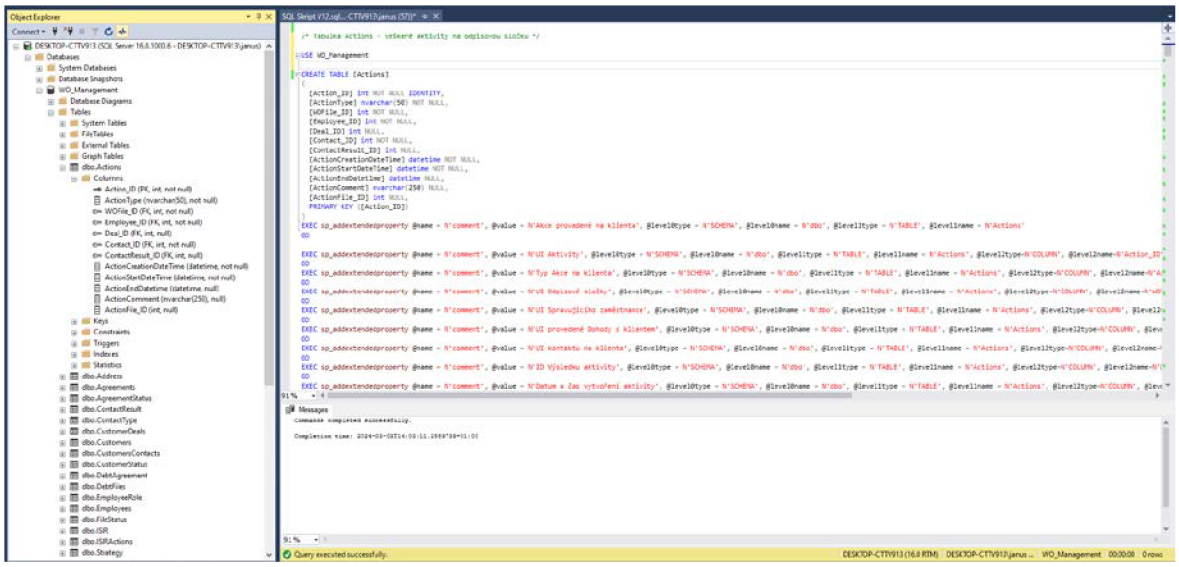
4.5.1. Testovací Software

Byl vybrán volně dostupný software od společnosti Microsfot Corporation SQL Server Management studio 19™, který umožňuje manipulaci s dostupnými servery. Pro vytvoření serveru WO_Management byl použit přidružený program SQL Server™ Configuration Manager 2022, na kterém byl server lokálně vytvořen.

4.5.2. Tvorba Databáze

Výše uvedené relace byly převedeny do skriptu SQL pro MS SQL Server™. Tento skript je v celém svém znění dostupný v příloze č. 2. Na obrázku 9 je viditelný průběh spuštění celého skriptu založení relací v databázi. Celková doba procesu byla 1 vteřina bez

reportovaných závad. To naznačuje správné vytvoření vazeb mezi tabulkami a řádnou strukturu skriptu. Po levé straně v sekci Object Explorer je viditelná definice jednotlivých klíčů a datových typů atributů tabulky Actions. Tato informace je velmi užitečná pro uživatele databáze.



Obrázek 9 - Vytvoření databáze WO_Management, zdroj: Vlastní zpracování

4.5.3. Import Dat

Dvěma hlavními ukazateli reportů oddělení vymáhání jsou celkové vybrané částky v měsíci a procentuální výběry na odpisovou složku v konkrétním měsíci. Tyto dva grafy budou vytvořeny v následující sekci. Naplněnými tabulkami proto budou WOfiles, u kterých sledujeme celkovou výši odepsané částky na složku, a Transactions. Ty zobrazují přijaté částky od klienta. Z důvodu vytvořených vazeb je třeba importovat data do všech tabulek, které mají vazby na testované relace. Proto musí být importovány data i do tabulek CustomerStatus, AgreementStatus, ContactType, FileStatus, ContactResult, Strategy, Employees, Teams, EmployeeRole a Customers. V příloze č. 3 je uveden celý skript importů dat do tabulek určených pro testovací případy. Čas nahrávání dat každé tabulky trval 1 vteřinu. Testovací data byla poskytnuta zadavatelskou firmou.

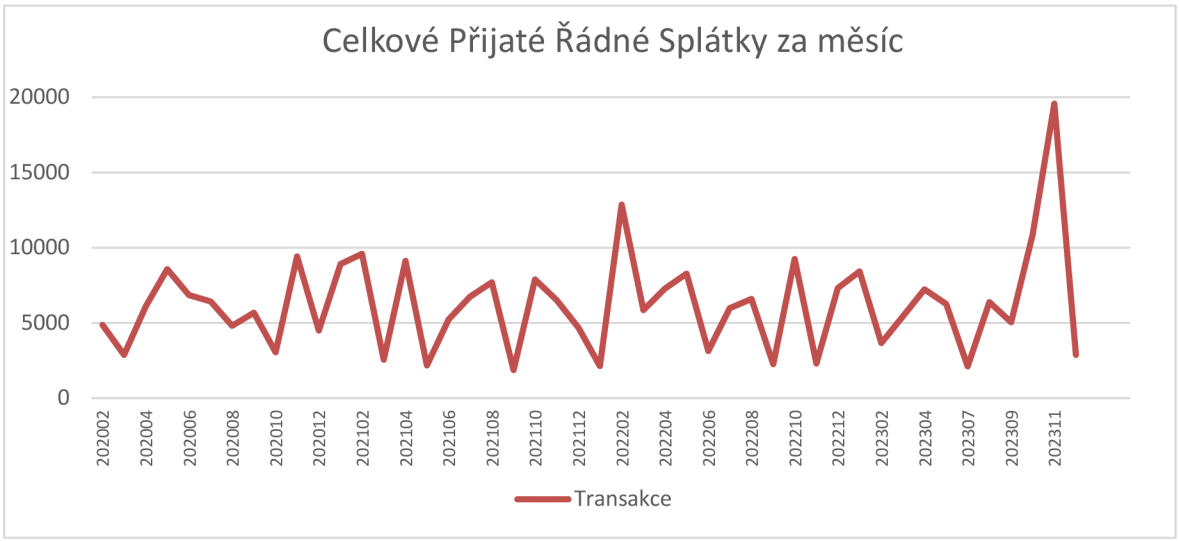
4.5.4. Testovací Reporty

Prvním testovacím grafem je přehled přijatých transakcí za jednotlivé měsíce. Částky se ověřují pouze za aktivní odpisové složky. Níže uvedený skript nám pomůže kýžená data z databáze získat. Pro lepší přehlednost v reportu byl uměle vytvořen atribut Month_ID, který reprezentuje číselné vyjádření roku a měsíce pozorování.

```sql
SELECT (cast(YEAR(Creationdate) as nvarchar) +
              CASE WHEN cast(MONTH(Creationdate) as nvarchar) < 10
                     THEN '0' + cast(MONTH(Creationdate) as nvarchar)
                     ELSE cast(MONTH(Creationdate) as nvarchar)
        END) as Month_ID
        , SUM(Transactionvalueczk) as Transakce
FROM Transactions A
INNER JOIN WOFiles B on a.WOFile_ID = B.WOFile_ID
WHERE TransactionType = 'Řádná splátka' -- vybíráme řádné splátky klienta
and B.WOFileStatus_ID =1 -- Vybíráme aktivní složky
GROUP BY
 (cast(YEAR(Creationdate) as nvarchar) +
              CASE WHEN cast(MONTH(Creationdate) as nvarchar) < 10
                     THEN '0' + cast(MONTH(Creationdate) as nvarchar)
                     ELSE cast(MONTH(Creationdate) as nvarchar)
        END)
ORDER BY Month_ID
```

(Zdroj: Vlastní zpracování)

Čas průběhu skriptu byl naměřen na jednu vteřinu. Z vytažených dat byl vytvořen společností žádaný graf, viz. obrázek



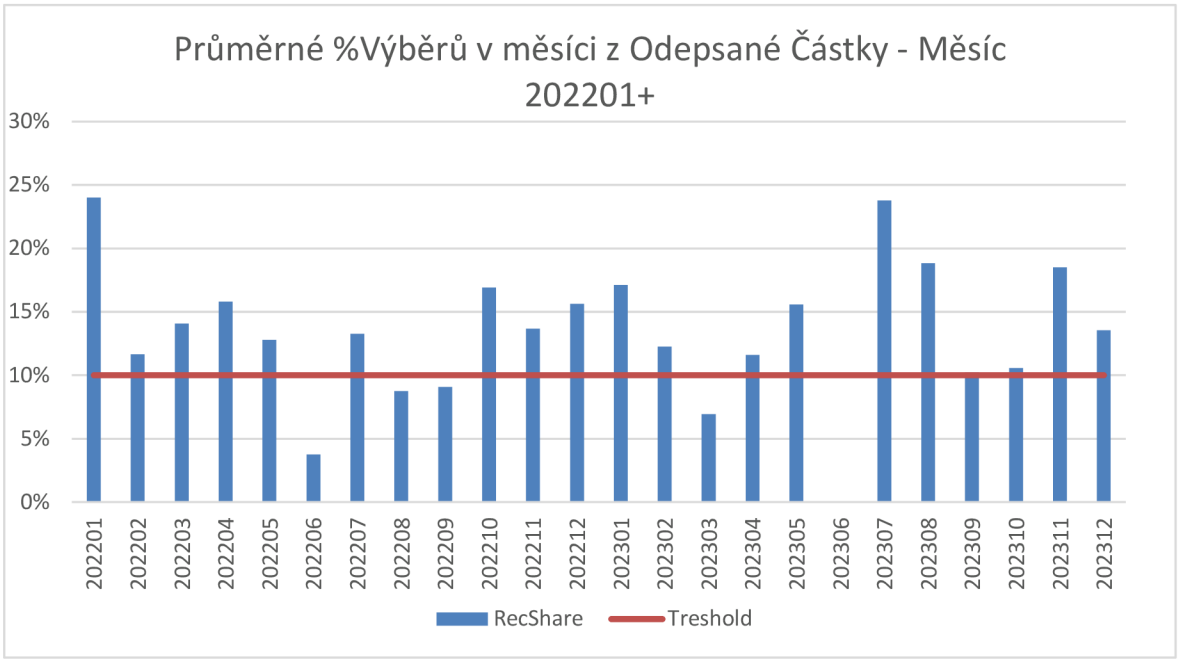
Obrázek 10 - Celkové přijaté splátky na složku měsíčně, zdroj: vlastní zpracování

Druhým požadovaným reportem je přehled procentuálních výběrů z odepsané částky na jednotlivé měsíce. Zajímají nás pouze měsíce od roku 2022 do současnosti. Další podmínkou je, že se opět musí jednat o aktivní odpisové složky a o řádné splátky. Skript pro požadovaná data je uveden níže.

```
SELECT Month_ID
,(AVG(cast(Recoveries as float)/(cast(WOValue as float)))) as RecShare
FROM
(
SELECT a.WoFile_ID, A.WOValue, SUM(ISNULL(B.Transactionvalueczk,0)) as
Recoveries,(cast(YEAR(Creationdate) as nvarchar) +
              CASE WHEN cast(MONTH(Creationdate) as nvarchar) < 10
                     THEN '0' + cast(MONTH(Creationdate) as nvarchar)
                     ELSE cast(MONTH(Creationdate) as nvarchar)
        END) as Month_ID -- Manuální vytvoření ID Měsíce
FROM WOFiles A
LEFT JOIN Transactions B ON a.WOFile_ID = B.WOfile_ID AND TransactionType = 'Řádná
splátka'
WHERE A.WoFileStatus_ID = 1
GROUP BY a.WoFile_ID, A.WOValue,(cast(YEAR(Creationdate) as nvarchar) +
              CASE WHEN cast(MONTH(Creationdate) as nvarchar) < 10
                     THEN '0' + cast(MONTH(Creationdate) as nvarchar)
                     ELSE cast(MONTH(Creationdate) as nvarchar)
        END)
) AS REC
WHERE Month_ID > 202200
GROUP BY Month_ID
       ORDER BY Month_ID
```

(Zdroj: Vlastní zpracování)

Čas průběhu skriptu byl reportován na jednu vteřinu. Z vybraných dat byl vytvořen graf, do kterého byl též zanesen tzv. *treshold,* neboli předpokládaná procentuální úroveň vymožených částek z celkového odpisu, na měsíc. Viz. obrázek 11.



Obrázek 11 - Průměrné procentuální výběry, zdroj: vlastní zpracování

5. Zhodnocení výsledků

V rámci studie byly vytvořeny teoretické základy pro tvorbu databáze ke správě odepsaných klientů. Veškeré postupy byly konzultovány se zástupci zadavatelské společnosti. Na základě těchto informací byl autorem práce vytvořen návrh relační databáze. Ten byl zpracován v SŘDB od společnosti Microsoft Corporation, MS SQL Server™. Tento software byl vybrán, jelikož se jedná o aplikaci využívanou zadavatelskou společností.

Na testovacích příkladech bylo ověřeno nejen vytvoření samotné databáze, ale také naplnění dat a funkčnost vzájemných vazeb mezi testovanými tabulkami. Doba zpracování přiložených skriptů nepřesáhla jednu vteřinu. Tím se potvrdila kvalita zpracovaného návrhu a jeho užitnost pro zmíněnou společnost.

Z Relační databáze je též možné vytvářet reporty, jak bylo prezentováno na konci praktické části. Databáze velmi dobře reaguje na vytvořené SQL skripty informace zobrazuje téměř okamžitě.

6. Závěr

Autor se v rámci práce zabývá problematikou odpisů a tvorbou návrhu relační databáze pro společnost spravující odepsané klienty. Prvním dílčím cílem byla analýza teoretického pozadí zmíněných témat. Tento cíl byl naplněn v teoretické části práce. V první sekci teoretické části byly popsány základní informace potřebné pro pochopení odpisů klientů z finančních institucí. Druhá sekce je zaměřena na obecný popis databází a východiska tvorby relační databáze.

Druhým dílčím cílem bylo vytvoření testovacího prostředí s daty. Toho bylo dosaženo v praktické části. Autor v příloze dodává skript potřebný pro vytvoření relační databáze v prostředí MS SQL Server™. Zároveň je doložen i SQL kód pro importování testovacích dat poskytnutých zadavatelskou společností.

Posledním dílčím cílem bylo ověřit funkčnost návrhu na testovacích datech. Data byla testována na dvou reportech požadovaných zmíněnou společností. Detail tohoto testu je poskytnut na konci praktické části, včetně přiloženého SQL skriptu.

Hlavním cílem práce bylo navržení funkčního modelu relační databáze pro správu odepsaných klientů. Tento model je vytvořen na začátku praktické části.

Hlavním přínosem této práce je zefektivnění a modernizace databáze firmy, která vymáhá klienty odepsané v mateřské společnosti a zároveň zavádí nový CRM systém. Model byl přizpůsoben požadavkům CRM aplikace, uživatelům výstupů databáze a datovým analytikům. Oproti původní databázi přináší nový návrh jasně definované vztahy mezi relacemi, popis jednotlivých atributů a jednodušší logiku pro práci v jazyce SQL. Návrh může sloužit i jako vzor případným čtenářům této práce pro jejich pracovní potřeby.

Dodatek závěrem. V době psaní této práce byla již ověřena funkčnost navrhované databáze ve firemním prostředí. Oddělení informačních technologií pracuje na tvorbě historických záznamů jednotlivých relací a projektový manažer společnosti dokončuje kroky potřebné pro přechod ze starého systému na nový.

7. Seznam použitých zdrojů

ČESKÁ NÁRODNÍ BANKA - *Spotřebitelský úvěr* [online] [vid. 2024-02-17]. Dostupné z: https://www.cnb.cz/cs/dohled-financni-trh/ochrana-spotrebitele/spotrebitelsky-uver/

ČESKÁ BANKOVNÍ ASOCIACE. Odpis úvěru. *Odpis úvěru* [online] [vid. 2024-02-11]. Dostupné z: https://cbaonline.cz/odpis-uveru

EXEKUTORSKÁ KOMORA ČR, 2024. Otevřená data o exekucích. *statistiky.ekcr.info* [online] [vid. 2024-02-11]. Dostupné z: http://statistiky.ekcr.info/statistiky

FRIEDRICHSEN, L. et al. 2020, *Concepts of Database Management.* 10. vyd. Boston: Cengage Learning Inc. 432 s. ISBN 978-0-357-42209-0.

CHLEBOVSKÝ, V. 2005. *CRM: řízení vztahů se zákazníky.* Vyd. 1. Brno: Computer Press. 190 s. ISBN 80-251-0798-1.

HOTEK, M. 2009. *Microsoft SQL Server 2008: krok za krokem.* Brno: Computer Press. 488 s. ISBN 978-80-251-2466-6.

INFO@AION.CZ, AION CS. 182/2006 Sb. Insolvenční zákon. *Zákony pro lidi* [online] [vid. 2024-02-18]. Dostupné z: https://www.zakonyprolidi.cz/cs/2006-182

MERUNKA, V.; VOSTROVSKÝ, V. 1998, *Databázové systémy.* Praha: Credit. 156 s. ISBN 80-213-0388-3.

MOLLINARO, A. *2009. SQL: kuchařka programátora.* Brno: Computer Press. 573 s. ISBN 978-80-251-2617-2.

POKORNÝ, J. 1992. *Databázové systémy a jejich použití v informačních systémech.* Praha: Academia. 313 s. ISBN 80-200-0177-8.

SOUKUP, R.; KRÁSENSKÝ, D. 1998. *Mistrovství v SQL serveru 6.5 : podrobný průvodce návrhem, implementací a optimalizací databází a architekturou SQL serverů.* Praha: Computer Press. 849 s. ISBN 80-7226-092-8.

STEPHENS, R. K. et al. 2010, *Naučte se SQL za 28 dní: [stačí hodina denně].* Brno: Computer Press. 728 s. ISBN 978-80-251-2700-1.

VESELÁ, J. et al. 2021. *Insolvence 2008-2020: data, názory, predikce*. Praha: Stálá konference českého práva, s. r. o. 288 s. ISBN 978-80-906813-3-0.

VOSTROVSKÝ, V. 2014. ČESKÁ ZEMĚDĚLSKÁ UNIVERZITA V PRAZE. KATEDRA INFORMAČNÍHO INŽENÝRSTVÍ. *Vytváření databází v ORACLE.* V Praze: Česká zemědělská univerzita, Provozně ekonomická fakulta. 132 s. ISBN 978-80-213-1191-6.

ZEDNÍČEK, J. 2024. Data | *Demokratizace a data driven přístup. Jan Zedníček - Data & Finance [online]*. [vid. 2024-02-18]. Dostupné z: https://janzednicek.cz/data-demokratizace-dat-a-data-driven-pristup/

8. Seznam obrázků, tabulek a zkratek

8.1.Seznam obrázků

8.2.Seznam tabulek

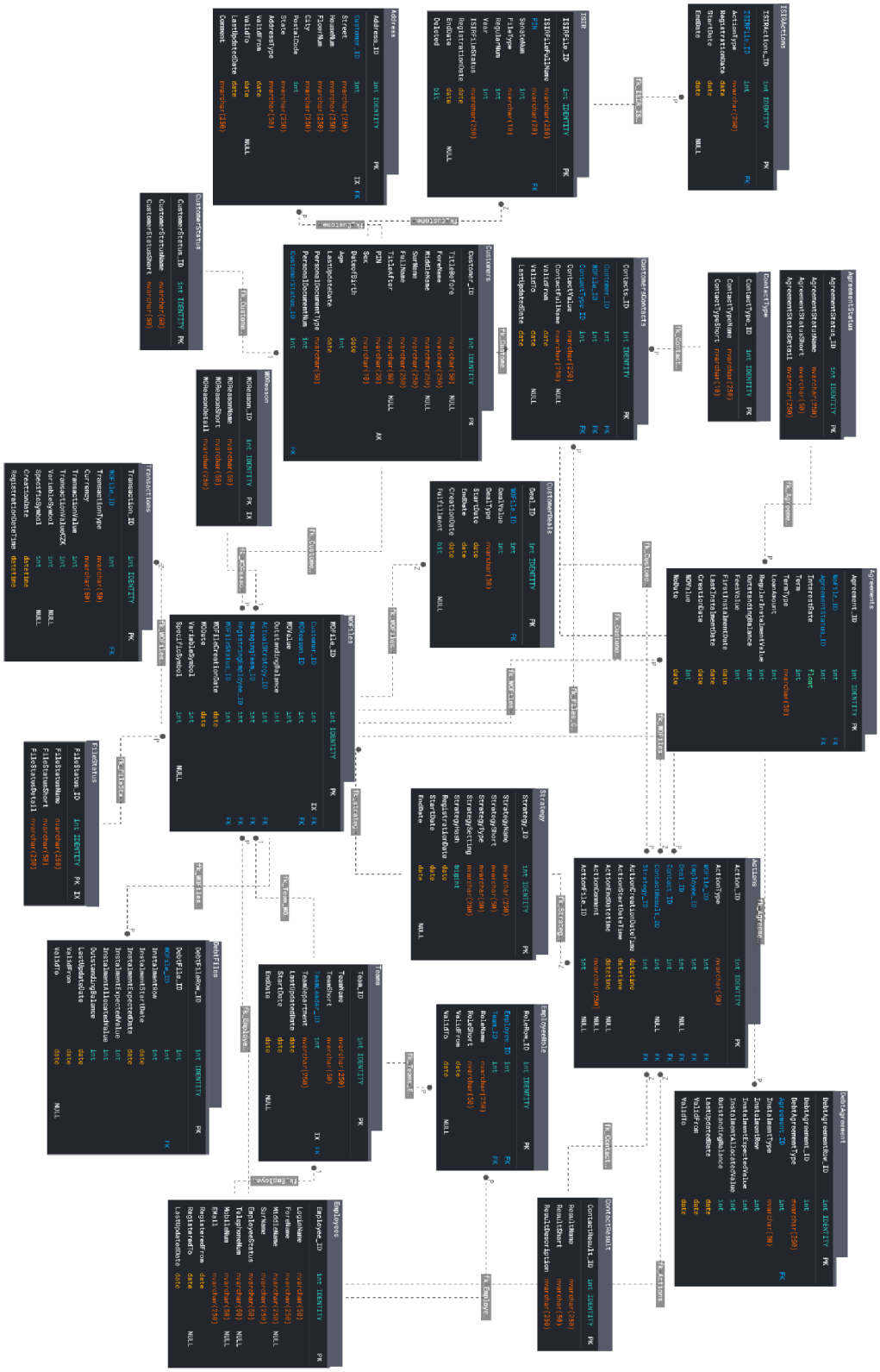
8.3.Seznam použitých zkratek

AI	Databází Generovaná Hodnota
AR	Arbitrary Key
CRM	Customer Relationship Management
CZK	Koruna Česká
ČNB	Česká Národní Banka
DB	Databáze

DBS	Databázové Systémy
DCL	Data Control Language
DDL	Data Definition Language
DML	Data Manipulation Language
DPD	Days Past Due
ERD	Entity Relationship diagram
ERM	Entity Relationship model
FK	Foreign Key
ID	Identification
ISIR	Insolvenční Informační Rejstřík
IX	Index
MS	Microsoft Corporation
NF	Normální Forma
OOPJ	Objektově Orientovaný Programovací Jazyk
SMS	Short Messaging Service
PK	Primary Key
SŘDB	Systém Řízení Báze Dat
SQL	Structured Query Language
TCL	Transaction Control Language
UI	Unikátní Identifikátor
WO	Write-Off
ZMT	Zaměstnanec

9. Přílohy

9.1. Příloha 1 – ER Diagram



9.2.Příloha 2 – Skript pro tvorbu relační databáze

https://github.com/Arcillion/Bakalarska_Prace/blob/main/Skript_Tvorba_Rela%C4%8Dn%C3%AD_Datab%C3%A1ze (Zdroj: Vlastní zpracování)

9.3.Příloha 3 – Skript pro vkládání testovacích dat

https://github.com/Arcillion/Bakalarska_Prace/blob/main/Skript_Insert_Testovac%C3%ADch_Dat (Zdroj: Vlastní zpracování)