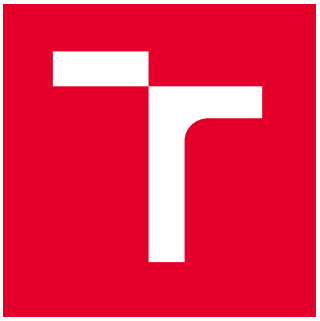


VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA PODNIKATELSKÁ

FACULTY OF BUSINESS AND MANAGEMENT

ÚSTAV INFORMATIKY

INSTITUTE OF INFORMATICS

ANALÝZA SOUČASNÉHO STAVU DATABÁZE A NÁVRH ŘEŠENÍ V KONKRÉTNÍ FIRMĚ

ANALYSIS OF THE CURRENT STATE OF THE DATABASE AND PROPOSAL OF A SOLUTION IN A SPECIFIC COMPANY

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — **Lukáš Valčík**
AUTHOR

VEDOUCÍ PRÁCE — **Ing. Jiří Kříž, Ph.D.**
SUPERVISOR

BRNO 2023

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA PODNIKATELSKÁ

Zadání bakalářské práce

Ústav:	Ústav informatiky
Student:	**Lukáš Valčík**
Vedoucí práce:	**Ing. Jiří Kříž, Ph.D.**
Akademický rok:	2022/23
Studijní program:	Manažerská informatika

Garant studijního programu Vám v souladu se zákonem č. 111/1998 Sb., o vysokých školách ve znění pozdějších předpisů a se Studijním a zkušebním řádem VUT v Brně zadává bakalářskou práci s názvem:

Analýza současného stavu databáze a návrh řešení v konkrétní firmě

Charakteristika problematiky úkolu:

Úvod
Cíle práce, metody a postupy zpracování
Teoretická východiska práce
Analýza současného stavu
Vlastní návrhy řešení
Závěr
Seznam použité literatury
Přílohy

Cíle, kterých má být dosaženo:

Cílem bakalářské práce je analýza stavu databázové struktury firmy a zlepšení stavu databázové struktury k zajištění efektivnějšího uchovávání dat.

Základní literární prameny:

JAŠEK, Petr. E-learningový kurs pro výuku jazyka SQL [online]. Brno, 2008 [cit. 2022-12-28]. Dostupné z: http://hdl.handle.net/11012/52950. Bakalářská práce. Vysoké učení technické v Brně. Fakulta informačních technologií. Ústav informačních systémů. Vedoucí práce Šárka Květoňová.

KOCH, Miloš a Bernard NEUWIRTH. Datové a funkční modelování. 3. vyd. Brno: Akademické nakladatelství CERM, 2008. 121 s. ISBN 978-802-1437-319.

SANJEEV, Dr. Verma. [online]. University of Lucknow,: Lucknow - 226021, 2020 [cit. 2022-12-28]. Dostupné z: https://www.lkouniv.ac.in/site/writereaddata/site Content/20200329162108510 1sanjeev_rdbms_unit-I_sql_bba_ms_4_sem.pdf

ŘÍHA, Petr. Návrh SQL databáze pro podporu činností malé IT firmy [online]. Brno, 2012 [cit. 2022-12-28]. Dostupné z: http://hdl.handle.net/11012/7828.

Termín odevzdání bakalářské práce je stanoven časovým plánem akademického roku 2022/23

V Brně dne 5.2.2023

L. S.

Ing. Jiří Kříž, Ph.D. garant	doc. Ing. Vojtěch Bartoš, Ph.D. děkan

Abstrakt

Bakalářská práce se zaměřuje na analýzu a návrh SQL databáze pro vybraný podnik se zaměřením na výrobu a dopravu. Obsahuje základní teoretické informace týkající se databázových systému, analýzu podniku a návrh databázové struktury pro vybraný podnik.

Abstract

The bachelor thesis focuses on the analysis and design of a SQL database for a selected company focusing on manufacturing and transport. It contains basic theoretical information regarding database systems, analysis of the enterprise and design of database structure for the selected enterprise.

Klíčová slova

Databáze, SQL, MS SQL Server, Data, Informace, Databázová struktura, Relace

Key words

Database, SQL, MS SQL Server, Data, Information’s, Database structure, relations

Bibliografická citace práce

VALČÍK, Lukáš. *Analýza současného stavu databáze a návrh řešení v konkrétní firmě* [online]. Brno, 2023 [cit. 2023-05-15]. Dostupné z: https://www.vutbr.cz/studenti/zav-prace/detail/152400. Bakalářská práce. Vysoké učení technické v Brně, Fakulta podnikatelská, Ústav informatiky. Vedoucí práce Ing. Jiří Kříž, Ph.D.

Čestné prohlášení:

Prohlašuji, že předložená bakalářská práce je původní a zpracoval jsem ji samostatně. Prohlašuji, že citace použitých pramenu je úplná, že jsem ve své práci neporušil autorská práva (ve smyslu Zákona č. 121/2000 Sb., o právu autorském a o právech souvisejících s právem autorským).

V Brně dne 15. 5. 2023 …….

Lukáš Valčík

Obsah

Úvod

Během studia na vybraných středních školách a vysokých školách či univerzitách se člověk učí práci s databázemi a její tvorbu. Problémem se poté stane, když tento student je přijat do firmy, aby zde spravoval jejich strukturu. Bohužel velká většina společností si nechala vytvořit strukturu a poté ji už dále neupravovala z pochopitelných důvodů (např. cenová tíže, časová náročnost či složitý přenos dat), ale společnosti, co mají databáze přeci starší jak 15 let, už nemohou odpovídat aktuálnímu stavu firmy. Společnosti se rozrůstají, mění se jejich cíle a konkurence. Konkurence je dneska asi největší problém. Pokud přijde nová firma a má k dispozici novou a lepší strukturu dat. je pak i jednodušší pro tuto firmu reálně fungovat.

Celý svět se v dnešní době zakládá na informacích. Informace čerpáme z dat, které musíme někde uchovávat. Časem se ve vývoji začali používat i uložiště zvané databáze kde můžeme uchovávat velké množství dat za relativně malou cenu s velice dobrým přístupem.

Důležitým faktorem v dnešním IT světě je bezpečnost. Ve světě, kde jsou informace nejcennějším zdrojem si musí firma svoje data chránit. Zastaralé databáze dříve neměli takové zabezpečení jako máme dnes, a i proto je potřeba, aby se ve firmách postupem času databáze přizpůsobovali.

Proto mým cílem v této bakalářské práci bude analyzovat stávají databázovou strukturu vybraného podniku a poté vytvořit vlastní návrh řešení nalezených problémů v analýze.

Vymezení problému a cíle projektu

Vymezení problému

Problémem, který budu v této bakalářské práci řešit, je zastaralost databázové struktury, nedostatečná bezpečnost staré databáze v reálné firmě (Společnost. Zabývající se výrobou, skladováním a prodejem pivních výrobků).

Cíle práce

Cílem bakalářské práce je analýza stavu databázové struktury firmy a zlepšení stavu databázové struktury k zajištění efektivnějšímu uchovávání dat.

1. Teoretická východiska práce

Tento oddíl je zaměřen na teoretické podklady nutné k zvládnutí této práce a získání informací k porozumění dané problematiky.

1.1. Informace

Informace lze shrnout jako fakta nebo znalosti předávané nebo získané o určitém předmětu nebo problému. Mohou mít mnoho různých podob, například písemné nebo tištěné materiály, zvukové nebo obrazové záznamy, data uložená v technologických zařízeních nebo počítačích, nebo dokonce konverzace mezi lidmi. [1]

Informace mají zásadní význam v různých oblastech života, včetně obchodu, výzkumu, technologií a vzdělávání. Efektivní shromažďování a zpracovávání informací je důležitým talentem, který může lidem pomoci při přijímání informovaných rozhodnutí, řešení problémů a celkově při zlepšování jejich blahobytu. [1]

Dnešní množství informací však může být také dezorientující a způsobovat informační přetížení. Abychom předešli informačnímu zahlcení a soustředili se na to, co je nejpodstatnější a nejdůležitější, je nezbytné pochopit, jak informace filtrovat a určovat jejich priority. [1]

1.2. Data

Data lze definovat jako soubor faktů, statistik nebo informací, které jsou uloženy nebo zpracovány počítačem nebo jiným elektronickým zařízením. Mohou mít podobu čísel, slov, obrázků nebo jiných typů médií. Data jsou obvykle generována z různých zdrojů, jako jsou senzory, zařízení, softwarové aplikace a interakce s lidmi. Shromážděná data mohou být nezpracovaná a nestrukturovaná, nebo mohou být uspořádána a zpracována do strukturovanějšího formátu, jako je databáze nebo tabulka. [1]

V dnešní digitální době objem generovaných dat exponenciálně roste a schopnost zpracovávat a analyzovat tato data je pro mnoho odvětví, včetně zdravotnictví, financí, marketingu a technologií, stále důležitější. [1]

Efektivní správa a analýza dat může poskytnout cenné poznatky a pomoci organizacím přijímat informovaná rozhodnutí. Je však důležité zajistit, aby data byla shromažďována, ukládána a zpracovávána bezpečným a etickým způsobem, aby bylo chráněno soukromí jednotlivců a zachována důvěra v rozhodování založené na datech. [1]

1.3. Datové modely

Datový model je obraz toho, jak jsou informace v systému nebo aplikaci organizovány, uchovávány a používány. Nastiňuje vazby mezi různými druhy dat a také předpisy, které řídí manipulaci s nimi a jejich ukládání. [2]

K tomu, aby bylo zaručeno, že data jsou přesná, konzistentní a dobře organizovaná, lze využít datové modely. Jsou užitečné zejména při návrhu databáze, kde mohou programátorům pomoci s vytvořením schématu databáze, které je efektivní a účinné a splňuje požadavky aplikace. [2]

Pro tvorbu datových modelů je k dispozici řada placených i neplacených nástrojů. Jako příklady jazyků pro modelování dat používáme modely entit a vztahů (Entity-Relationship, ER), Unified Modeling Language (UML) a Business Process Modeling Notation (BPMN). [1]

Lineární datové modely dělíme na:

Lineární datový model

Relační datový model

Síťový datový model

Hierarchický datový model

1.3.1. Lineární datový model

Lineární datový model je snadno pochopitelný datový formát, který zobrazuje data jako seznam položek v pořadí za sebou. Ukazatel nebo odkaz spojuje každý člen seznamu s dalším prvkem a vytváří lineární řetězec dat. [1]

Pokud je třeba data načítat postupně, například v propojeném seznamu nebo frontě, je tato datová architektura často využívána. Efektivní procházení propojeného seznamu je umožněno tím, že každý člen má odkaz na následující člen. [2]

Jednoduchost a snadnost použití lineárního datového modelu jsou dvě jeho hlavní výhody. Lze k němu použít základní programovací nástroje, jako jsou pole, ukazatele a reference. Jednoduchou změnou příslušných ukazatelů nebo referencí lze lineární datový model také snadno aktualizovat tak, aby zahrnoval nebo vylučoval nové položky. [1]

Lineární datový model má však určité nevýhody. Protože prohledávání celého seznamu za účelem nalezení konkrétního člena může být neefektivní, nehodí se dobře pro scénáře, kde je nutný náhodný přístup k datům. Lineární datový model je navíc méně efektivní při popisu složitějších datových struktur, protože neumožňuje složité interakce mezi datovými komponentami. [1]

1.3.2. Hierarchický datový model

Nejstarší databázový model je tento. Vznikl v 60. letech 20. století. Pomocí něj lze popsat všechny informační systémy nebo jejich části, které mají vztahy podřízenosti a nadřízenosti. Tento přístup však nedokáže dostatečně vysvětlit mnoho informačních systémů. Pomocí tohoto paradigmatu například nelze definovat všechny myslitelné vazby (vztahy) mezi věcmi. [2]

Vzhledem k tomu, že tento model nebyl odborníky a výzkumníky uznán, byl rychle vytvořen nový databázový model. [1]

Informační systém pro rodokmen je příkladem systému, který lze reprezentovat pomocí hierarchického modelu. [2]

1.3.3. Síťový model

Jedná se o model, který je uspořádán do grafu podle jednotlivých entit. Matematická struktura složená z uzlů a hran se nazývá graf. Uzly informačního systému jsou jeho jednotlivé entity a hrany, které je spojují, představují vytvořené vazby mezi těmito entitami. Tento model tedy může buď zaznamenávat vazby, které nejsou hierarchické (jeden objekt je nadřazen jiné entitě), neboť je již podstatně širší než hierarchický model. Přesto je tento model stále nedostatečný nebo není schopen plně zachytit všechny vazby, zejména ty, v nichž se entity chovají vícenásobně. [2]

V důsledku toho se od tohoto modelu postupně upouští a vytváří se konečný tedy relační model, který je dodnes hojně využíván. [2]

Některé geografické informační systémy (GIS) lze popsat pomocí síťového modelu.

1.3.4. Relační datový model

Jedná se o poslední model databáze, který se stále běžně používá. Relace (tabulka) je základním stavebním prvkem. Relace popisuje každou entitu v informačním systému a také každé spojení, které entity propojuje. Soubor relací se nazývá relační schéma. [2]

Vývoj relačního modelu souvisí s vývojem jazyka SQL, který se vyvinul v průmyslový standard pro relační databáze. Jeho původní verze SQL92 z let 1986-1992 byla nakonec nahrazena podstatně lepší verzí. V některých databázových systémech se tato konkrétní verze dodnes často používá. [1]

1.4. Databáze

1.4.1. Historie před databázemi

Před rozvojem počítačových databází se informace obvykle ukládaly a spravovaly manuálními nebo analogovými metodami. V počátcích lidské civilizace se informace zaznamenávaly na kamenné desky, papyrové svitky a další formy raného písma. Tyto rané záznamy měly omezený rozsah a často sloužily k účetním účelům nebo k evidenci důležitých událostí.[4]

Jak se společnosti stávaly složitějšími, objevily se nové formy záznamů, včetně hliněných tabulek, voskových tabulek a pergamenových svitků. Tyto záznamy se používaly k různým účelům, včetně sledování hospodářských transakcí, zaznamenávání náboženských rituálů a uchovávání historických událostí. .[3]

Ve středověku byli k vytváření a vedení záznamů zaměstnáváni písaři, kteří používali brkové pero a inkoust na pergamenu nebo papíře. Tyto záznamy se často uchovávaly v knihovnách nebo archivech a používaly se pro právní, náboženské a administrativní účely.

S vynálezem knihtisku v 15. století se rozšířily knihy a další tištěné materiály a systémy vedení záznamů se staly organizovanějšími a systematičtějšími. Avšak teprve s rozvojem počítačů a digitálních technologií vznikly moderní databázové systémy, které umožnily efektivní ukládání a vyhledávání obrovského množství dat. Začátky dnešních databází sahají do počátků výpočetní techniky. V 60. letech 20. století byl vyvinut nový typ počítačového systému zvaný mainframe a podniky a organizace je začaly používat k ukládání velkého množství dat. Data však byla často uložena ve formátu plochých souborů, což ztěžovalo vyhledávání a získávání informací. .[4]

1.4.2. Historie moderních databází

Počátkem 70. let 20. století vznikla nová třída databázových systémů známá jako hierarchická databáze. Každá část dat v tomto systému byla uspořádána do stromové struktury s vazbou rodič-dítě. Hierarchickou databázi hojně využívaly bankovní a finanční sektory, protože umožňovala rychle a efektivně vyhledávat údaje o účtech klientů.

Na přelomu 70. a 80. let 20. století vznikl nový typ databáze známý jako relační databáze. Každá tabulka v organizaci dat tohoto systému se skládala z řádků a sloupců. Hlavní dnes používaný databázový systém se rychle vyvinul z relační databáze, protože podporoval složité dotazy a propojení mezi tabulkami. .[4]

S rozvojem internetu a elektronického obchodování v 90. letech 20. století byly vytvořeny nové databázové systémy, které měly spravovat obrovské objemy dat.

V současné době se databázové technologie dále vyvíjejí s růstem objemu dat a internetu věcí (Internet of Things), protože organizace se snaží ukládat a zpracovávat stále větší objemy dat. Pro řešení specifických potřeb správy dat se vyvíjejí nové typy databází, jako jsou grafové databáze a databáze časových řad. .[4]

1.4.3. Databázové pojmy

Níže uvedené pojmy jsou základem pro používání databází. Některé pojmy odpovídají určitým pojmům z reálného světa.

1.4.3.1. Entita

Je základním kamenem celé struktury. V reálném světě to reprezentuje jakoukoliv věc, kterou můžeme popsat. (např. člověk, škola, dům atd.)

1.4.3.2. Atribut

Atributy jsou jakýkoliv vlastnosti Entit. Takže pokud vezmeme člověka, tak jeho atributy budou např. barva očí, bydliště, zaměstnání, matka atd.

1.4.3.3. Entitní vazby

Entity mají mezi sebou nějaké vztahy, které nazýváme relace či vazby. V reálném světě by tyto vztahy najdeme např. Člověk a Matka jejich vztah je pak dále popsán podle určení druhu relace. Určíme si entitu A, a entitu B. Relace se vždy určují tak že si vezmeme jednu entitu A, a tu vztahem porovnáváme s entitou B, tak dostaneme jednu stranu vazby. Abychom

získali druhou stranu vazby musíme to obrátit a vzít si jednu entitu B a porovnat jí s entitou A.[4]

- Relace 1:1 - Těchto relací se ve světě moc neobjevuje. Jako příklad můžeme zvolit firmu. Firma musí mít vždy a pouze jen jedno sídlo. Tím pak víme že jedna firma může mít pouze jedno sídlo tedy 1:1

- Relace 1:* - Tyto relace spolu s *:1 jsou nejčastějším případem. Jako příklad můžeme uvést měsíce u planet. Každá 1 planeta může mít několik měsíců, ale zároveň měsíc má vždy jen jednu planetu, kolem které krouží

- Relace *:* - Tato vazba je jedním z problému v relačních databázích. V další části této práce se tomuto problému budeme věnovat. Příkladem této relace je že 1 kantor může vyučovat více předmětů, ale zároveň jeden předmět může vyučovat více kantorů

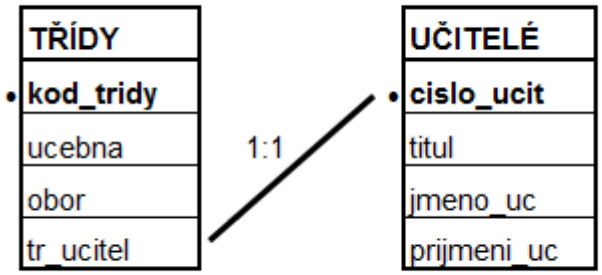
1.4.3.4. Primární klíč

Primární klíč je jedinečný identifikátor každého záznamu nebo řádku v tabulce. Je to sloupec nebo kombinace sloupců, které slouží jako jedinečný identifikátor pro konkrétní záznam v tabulce.

Pomocí primárního klíče může databáze rychle a efektivně vyhledávat a aktualizovat jednotlivé záznamy v tabulce. Může také pomoci zajistit přesnost a konzistenci dat tím, že zabrání přidání duplicitních nebo neúplných záznamů do tabulky. Primární klíč lze navíc použít jako referenční nebo cizí klíč v jiných tabulkách a vytvořit tak vztahy mezi různými sadami dat v databázi.[4]

1.4.3.5. Cizí klíč

Cizí klíč je sloupec nebo kombinace sloupců v tabulce, která odkazuje na primární klíč jiné tabulky. Vytváří vztah mezi dvěma tabulkami a umožňuje sdílení a propojení dat mezi tabulkami.



Obrázek 1 - Spojení pomocí cizího klíče zdroj: vlastní zpracování

Zde vidíme. Že primárním klíčem v tabulce třídy je *kod_tridy* a v tabulce učitelé je to *cislo_ucit.* Jako cizí klíč zde je tr_ucitel. Třídní učitel může mít vždy jen jedno číslo učitele proto tedy vazba 1:1.

1.5.Normalizace

Normalizace dat v databázi snižuje redundanci a zvyšuje integritu dat. Cílem normalizace je zabránit nekonzistencím a abnormalitám v datech, ke kterým může dojít, pokud jsou data duplikována nebo uchovávána v nestandardním formátu. .[3]

Aby bylo možné vytvořit asociace mezi tabulkami pomocí primárních a cizích klíčů, znamená normalizace rozdělení databáze na menší, lépe zvládnutelné části známé jako tabulky. Tento postup se často skládá z několika fází známých jako normální formy, z nichž každá nastiňuje konkrétní kritéria, která musí být splněna, aby byla databáze považována za normalizovanou. Pro udržovaní normalizace používáme normálové formy (NF) .[5]

1.Normální forma (1NF) - Relace je v první normální formě, pokud každý její atribut obsahuje jen atomické hodnoty. Tedy hodnoty z pohledu databáze již dále nedělitelné.

2.Normální forma (2NF) - Databáze je ve druhé normální formě, pokud je v první normální formě a pokud žádný neklíčový atribut není závislý pouze na části klíče. Jinými slovy to znamená, že pokud chceme znát hodnotu nějakého atributu, neměla by stačit pouze znalost části klíče daného záznamu. Tento problém automaticky odpadá v případě, že má entita pouze jednoduchý klíč. V tom případě je databáze ve druhé normální formě ihned, jakmile je v první normální formě.

3.Normální forma (3NF) - Relace je ve třetí normální formě, pokud je ve druhé normální formě, a navíc všechny její neklíčové atributy jsou vzájemně nezávislé

4. Boyce-Codd normální forma (BCNF) - BCNF minimalizuje redundance a zvyšuje integritu dat. BCNF byl vytvořen jako rozšíření třetího normálního tvaru, nebo 3NF, v roce 1974 Raymond Boyce a Edgar Codd. Pracovali na vytvoření databázových schémat, které minimalizují propouštění s cílem snížit výpočetní čas. Třetí normální formulář odstraňuje sloupce, které nejsou závislé na primárním klíči, kromě toho, že splňují pokyny v první a druhé normální formě. BCNF, který je někdy označován jako 3.5NF, splňuje všechny

požadavky 3NF a vyžaduje, aby kandidátní klíče neměly závislost na jiných atributech v tabulce. .

1.6.Integrita databáze

Integrita databáze znamená, že data v ní uložená jsou konzistentní vůči definovaným pravidlům. Lze zadávat pouze data, která vyhovují předem definovaným kritériím (např. musí respektovat datový typ nastavený pro daný sloupec tabulky, či další omezení hodnot přípustných pro daný sloupec). K zajištění integrity slouží integritní omezení. Jedná se o nástroje, které zabrání vložení nesprávných dat či ztrátě nebo poškození stávajících záznamů v průběhu práce s databází. Například je možné zajistit mazání dat, která již ztratila svůj význam – například smažeme-li uživatele, odstraní se i zbytek jeho záznamů v ostatních databázových tabulkách. .[5]

Druhy integritních omezení

Pravidla pro údržbu integrity relací:

- Doménová integrita – Doména = obor hodnot = množina všech přípustných hodnot daného atributu
- Přechodová integrita – definování stavů, kterými může vektor hodnot právoplatně přecházet
- Entitová integrita – zabezpečuje integritu entit modelovaných v daném systému tzn. Každá entita musí být jednoznačně identifikovatelná –

Zajištění zachování vztahů mezi relacemi

- Referenční integrita – je nástroj, který pomáhá udržovat vztahy mezi záznamy v relačně propojených tabulkách. Pravidlo referenční integrity vyžaduje, aby pro každý záznam v podřízené tabulce, jehož klíč nemá hodnotu NULL, existoval v nadřízené tabulce záznam se stejným klíčem.[6]

Kontrola způsobu manipulace s daty

- Transakční integrita – Definuje přípustné způsoby manipulace s databází • Na rozdíl od ostatních omezení jsou tato omezení procedurálního charakteru a nejsou součástí datového modelu

1.7. Relační databáze

Data jsou v relačních databázích uchovávána v tabulkách se zavedenými schématy, kde každá tabulka představuje určitou entitu nebo myšlenku a každý řádek tabulky představuje jednu instanci této věci. Každá buňka v tabulce má jednu hodnotu, přičemž řádky označují vlastnosti dané entity a sloupce její atributy. .[6]

Schopnost vytvářet vztahy mezi tabulkami je jednou z hlavních vlastností relačních databází. Toho se dosahuje pomocí klíčů, které slouží jako odlišné identifikátory pro každý řádek tabulky. Každý řádek v tabulce je jednoznačně identifikován svým primárním klíčem, zatímco vazby mezi tabulkami se vytvářejí pomocí cizích klíčů.[7]

Data lze propojovat a přistupovat k nim z různých tabulek pomocí cizích klíčů, které jsou primárními klíči v jiných databázích.

Relační databáze používají k dotazování a správě dat strukturovaný dotazovací jazyk (SQL). Uživatelé mohou provádět operace, jako je vyhledávání dat, vkládání dat, aktualizace dat a mazání dat. Jazyk SQL, který nabízí standardizovaný způsob interakce s relačními databázemi.

Mezi výhody používání relačních databází patří konzistence a integrita dat, podpora složitých datových spojení a schopnost provádět efektivní vyhledávání a agregace na obrovských souborech dat. Relační databáze však mohou být omezeny problémy s rychlostí při práci s velmi velkými soubory dat nebo vysokým počtem současně pracujících uživatelů. .[7]

Celkově lze říci, že relační databáze byly široce používány a nadále zůstávají oblíbenou volbou pro ukládání a správu strukturovaných dat v různých aplikacích, od podnikových aplikací po webové aplikace.

1.8. ETL

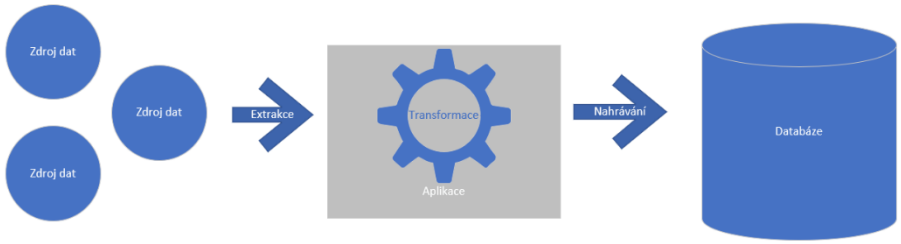
ETL je zkratka pro Extract(Extrakce), Transform(Transformace) a Load(nahrávání), což je proces používaný k integraci dat z různých zdrojů do cílového systému.

Extraction: Extrakce dat se provádí z různých zdrojů, včetně databází, tabulek a online služeb. Kroky tohoto procesu jsou vyhledání relevantních dat, výběr zdrojů dat a extrakce dat do formátu pro další zpracování.

Transformation: Aby bylo zajištěno, že extrahovaná data jsou správná, komplexní a konzistentní, zahrnuje krok transformace jejich čištění a změnu. Vedle všech nezbytných konverzí nebo výpočtů dat zahrnuje tento postup také validaci dat, obohacení dat a normalizaci dat.

Load: Během fáze načítání se převedená data načítají do cílového systému, například do datového skladu. V průběhu této fáze jsou data mapována. Tento postup zahrnuje ověření dat, jejich vložení do cílového systému a mapování do systému.

ETL je klíčovým krokem v procesu integrace dat, protože umožňuje podnikům míchat data z mnoha zdrojů, získávat z nich poznatky a využívat je k řízení strategických rozhodnutí. Postupem času se technologie ETL vyvíjely a novější systémy nyní nabízejí sofistikovanější funkce včetně integrace dat v reálném čase, sledování kvality dat a správy dat.



Obrázek 2 - ETL zdroj: vlastní zpracování

1.9.Jazyk SQL

Správa relačních databází se často provádí pomocí programovacího jazyka SQL (Structured Query Language). Pomocí této standardizované a účinné metody interakce s databázemi mohou uživatelé provádět akce včetně přístupu k datům, jejich vkládání, aktualizace a mazání.[8]

Relační model, na kterém je jazyk SQL postaven, uspořádává data do tabulek s řádky a sloupci. Každý sloupec v tabulce v databázi SQL odpovídá atributu nebo vlastnosti entity a každý řádek jedinečné instanci této entity podle určených schémat. Ke změně dat v těchto tabulkách nabízí SQL řadu příkazů a procedur. [8]

1.9.1. Základní SQL příkazy

SELECT: Tento příkaz se používá k získání dat z jedné nebo více tabulek v databázi. Umožňuje uživatelům zadat sloupce, které chtějí načíst, filtrovat data na základě podmínek, spojovat více tabulek a třídit data. [8]

INSERT: Tento příkaz slouží k vložení nových dat do tabulky. Uživatelé mohou zadat hodnoty, které chtějí vložit do jednotlivých atributů, nebo použít poddotazy pro vložení dat z jiné tabulky. [8]

UPDATE: Tento příkaz se používá k úpravě stávajících dat v tabulce. Uživatelé mohou zadat hodnoty, které chtějí aktualizovat v jednotlivých atributech, a pomocí podmínek filtrovat řádky, které chtějí aktualizovat.[8]

DELETE: Tento příkaz slouží k odstranění dat z tabulky. Uživatelé mohou zadat podmínky pro filtrování řádků, které chtějí odstranit.

CREATE: Používá se k vytváření nových objektů

ALTER: Využívá se ke změně či úpravě existujících objektů

DROP: Pomocí této funkce odstraňujeme objekty

Níže vám popíšu jednoduchý výběrový příkaz SELECT z databáze, kterou blíže popíšu později

```
SELECT FirstName, SurName, sLogName, sGroup, sName, sDescription FROM dbo.FlowerUser
join dbo.FlowerUserFunction on
dbo.FlowerUser.ID_User = dbo.FlowerUserFunction.ID_User
join dbo.FlowerFunction on
dbo.FlowerUserFunction.ID_Function = dbo.FlowerFunction.ID_Function
```

100 %

Results | Messages

	FirstName	SurName	sLogName	sGroup	sName	sDescription
1	Obsluha	Obsluha	user	Systém	Události systému	Nahlížení na log systémových událostí
2	Obsluha	Obsluha	user	Systém	Události systému	Nahlížení na log systémových událostí
3	Obsluha	Obsluha	user	Systém	Události systému	Nahlížení na log systémových událostí
4	Obsluha	Obsluha	user	Systém	Události systému	Nahlížení na log systémových událostí
5	Obsluha	Obsluha	user	Systém	Spravovat nastavení funkcí	Přístup ke správě funkcí v systému
6	Obsluha	Obsluha	user	Systém	Spravovat nastavení funkcí	Přístup ke správě funkcí v systému
7	Obsluha	Obsluha	user	Systém	Spravovat nastavení funkcí	Přístup ke správě funkcí v systému
8	Obsluha	Obsluha	user	Systém	Spravovat nastavení funkcí	Přístup ke správě funkcí v systému
9	Obsluha	Obsluha	user	Systém	Spravovat nastavení rolí	Přístup ke správě rolí
10	Obsluha	Obsluha	user	Systém	Spravovat nastavení rolí	Přístup ke správě rolí
11	Obsluha	Obsluha	user	Systém	Spravovat nastavení rolí	Přístup ke správě rolí
12	Obsluha	Obsluha	user	Systém	Spravovat nastavení rolí	Přístup ke správě rolí
13	Obsluha	Obsluha	user	Systém	Spuštění aplikace	Spuštění aplikace
14	Obsluha	Obsluha	user	Systém	Spuštění aplikace	Spuštění aplikace
15	Obsluha	Obsluha	user	Systém	Spuštění aplikace	Spuštění aplikace
16	Obsluha	Obsluha	user	Systém	Spuštění aplikace	Spuštění aplikace
17	Obsluha	Obsluha	user	Systém	Editace uživatelů	Editace uživatelů
18	Obsluha	Obsluha	user	Systém	Editace uživatelů	Editace uživatelů
19	Obsluha	Obsluha	user	Systém	Editace uživatelů	Editace uživatelů
20	Obsluha	Obsluha	user	Systém	Editace uživatelů	Editace uživatelů
21	Obsluha	Obsluha	user	Trendy	Správa skupin	Přístup ke správě skupin proměnných
22	Obsluha	Obsluha	user	Trendy	Správa skupin	Přístup ke správě skupin proměnných
23	Obsluha	Obsluha	user	Trendy	Správa skupin	Přístup ke správě skupin proměnných
24	Obsluha	Obsluha	user	Trendy	Správa skupin	Přístup ke správě skupin proměnných
25	Obsluha	Obsluha	user	Trendy	Správa pohledů	Přístup ke správě pohledů proměnn...
26	Obsluha	Obsluha	user	Trendy	Správa pohledů	Přístup ke správě pohledů proměnn...
27	Obsluha	Obsluha	user	Trendy	Správa pohledů	Přístup ke správě pohledů proměnn...
28	Obsluha	Obsluha	user	Trendy	Správa pohledů	Přístup ke správě pohledů proměnn...

Obrázek 3 Dotazovací skript zdroj: vlastní zpracování

SELECT [Sloupce, které chceme, aby byly vypsány ve výstupu] FROM [Tabulka výběru]

JOIN [Databáze].[Přidávaná tabulka] ON

[Databáze].[Tabulka výběru].[Atribut]=[Databáze].[Přidávaná tabulka].[Atribut]

1.9.2. Agregační funkce v SQL

V této části se budu věnovat zabudovaným funkcím v jazyku SQL, které nám umožňují shrnovat data, které poté můžeme později použít pro statistické zpracování. Nejdříve si ale musíme představit základní výrazy pro manipulaci s daty tedy GROUP BY a ORDER BY, kde první zmiňovaný nám sloučí všechny řádky podle stejných hodnot jednoho či více atributů, přičemž ORDER BY seřadí data. Pokud chceme, aby byla data seřazená vzestupně, musíme přidat příznak ASC a naopak, pokud chceme, aby data byla sestupně seřazená, musíme přidat příznak DESC. [8]

```
SELECT atribut
FROM tabulka
ORDER BY atribut ASC|DESC;
```

Teď, když už jsme si představili GROUP BY a ORDER BY vám můžu blíže popsat základní agregační funkce. V jazyce SQL totiž nemůžeme použít agregační funkci bez toho, abychom sloučili na konci dotazu data v řádcích. Mezi nejpoužívanější agregační funkce patří AVG(), SUM(), COUNT(), MIN(), MAX().[8]

AVG – Tato funkce vytvoří průměr zadaného číselného atributu

SUM – Vrátí součet tedy sumu zadaného číselného atributu

COUNT – Vrátí počet výskytů přesněji počet řádků, ve kterých se data nachází

MIN – Vrátí minimální hodnotu v zadaném atributu

MAX – Vrátí maximální hodnotu v zadaném atributu

1.9.3. Datové typy

datové typy dělíme na:

- číselné(numerické):

  Tyto čísla dále dělíme na

  - Přesná čísla:
    - BIGINT - 8 bajtů
    - INT - 4 bajty
    - SMALLINT - 2 bajty
    - TINYINT - 1 bajt
    - BIT/BOOLEAN – 0/1 PRAVDA/NEPRAVDA
    - DECIMAL (maximální počet desetinných míst)
    - MONEY – Pouze dvě desetinná místa, až 8 bajtů
  - Přibližná čísla:
    - FLOAT (plovoucí desetinná čárka) až 9 bajtů
  - Datum a čas:
    - DATE – Ve formě (RRRR-MM-DD)
    - TIME – Ve formě (HH:MM:SS:[NNNNNNN])
    - DATETIME – Ve formě (RRRR-MM-DD HH:MM:SS:[NNN])
  - Řetězce (text):
    - CHAR(n) – pevná délka n. v paměti rezervováno n bajtů
    - VARCHAR(n/max) – variabilní délka od n po max

1.10. MS SQL Server

Společnost Microsoft Corporation vytvořila systém pro správu relačních databází (RDBMS) známý jako Microsoft SQL Server. Jedná se o jeden z nejoblíbenějších a nejpoužívanějších systémů správy databází na světě, který využívají podniky všech velikostí a v různých odvětvích.[9]

MS SQL Server je známý svou stabilitou, zabezpečením a škálovatelností. Lze v něm zpracovávat velké objemy dat a nabízí vysoký výkon a spolehlivost. Podporovány jsou jak úlohy OLTP (online zpracování transakcí), tak OLAP (online analytické zpracování). [9]

MS SQL Server má řadu důležitých funkcí, mezi které patří např:

Integrace dat: Podporována je řada možností integrace dat, včetně replikace dat, importu/exportu a integrace s jinými zdroji dat.

Zabezpečení: Nabízí několik bezpečnostních opatření, jako je šifrování, ověřování a řízení přístupu, pro ochranu dat.

Business intelligence: V tomto případě se jedná o skupinu technologií, které firmám umožňují získávat informace z jejich dat, jako jsou služby pro reporting a analýzu.

Vysoká dostupnost: V zájmu zajištění vysoké dostupnosti dat a omezení výpadků obsahuje funkce, jako je clustering a replikace.

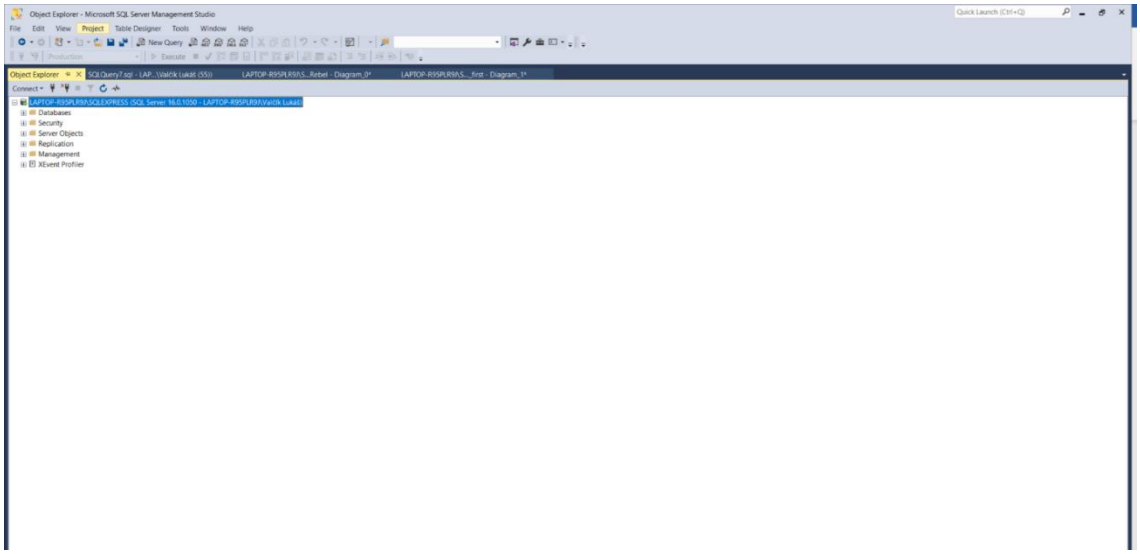
Závěrem lze říci, že MS SQL Server je robustní a důvěryhodný systém správy databází, který se často používá v podnikovém prostředí ke zpracování důležitých dat a aplikací.

1.11. MS Management studio

Ke správě a dohledu nad instancemi a databázemi SQL Serveru se používá nástroj grafického uživatelského rozhraní Microsoft SQL Server Management Studio (SSMS). Jedná se o bezplatný program nabízený společností Microsoft, který lze stáhnout jako součást instalačního disku pro SQL Server. [9]

SSMS umožňuje spravovat objekty Analysis Services, například provádět zálohování a zpracovávat objekty.

Management Studio poskytuje projekt Analysis Services Script, ve kterém můžete vyvíjet a ukládat skripty napsané v jazycích MDX (Multidimensional Expressions), DMX (Data Mining Extensions) a XML for Analysis (XMLLA). Pomocí projektů Analysis Services Scripts můžete provádět úlohy správy nebo znovu vytvářet objekty, například databáze a kostky, v instancích Analysis Services. V projektu Analysis Services Script můžete například vytvořit skript XMLA, který vytvoří nové objekty přímo na existující instanci Analysis Services. Projekty Analysis Services Scripts lze uložit jako součást řešení a integrovat je se správou zdrojového kódu. [10]



Obrázek 4 - MS Management studio zdroj: vlastní zpracování

2. Analýza současného stavu

Následující kapitola představuje společnost, pro kterou bude navržena databáze. Prvně představím samotný podnik a uvedu základní informace. Dále vám blíže popíšu organizační strukturu podniku. Dále se budu věnovat hardwarovému a softwarovému vybavení firmy a poté samotné analýze databázové struktury podniku.

2.1.Základní informace o společnosti

Data, ze kterých tato práce vychází jsou z reálné firmy, jen si ředitel firmy nepřeje, aby byla firma někde jmenována či aby bylo přímo poznat, o jakou firmu se jedná. Proto některé informace budou smyšlené.

Název společnosti: XXX Beer Brewery a.s.

Sídlo: Česká republika

Datum vzniku: v 19. století

klasifikace činností: Výroba průmyslových krmiv pro hospodářská zvířata, destilace, rektifikace a míchání lihovin, výroba piva, výroba sladu, zprostředkování velkoobchodu a velkoobchod v zastoupení, nespecializovaný velkoobchod, ostatní ubytování, stravování v restauracích, u stánků a v mobilních zařízeních, poradenství v oblasti řízení, reklamní činnosti, pronájem a leasing výrobků pro osobní potřebu a převážně pro domácnost

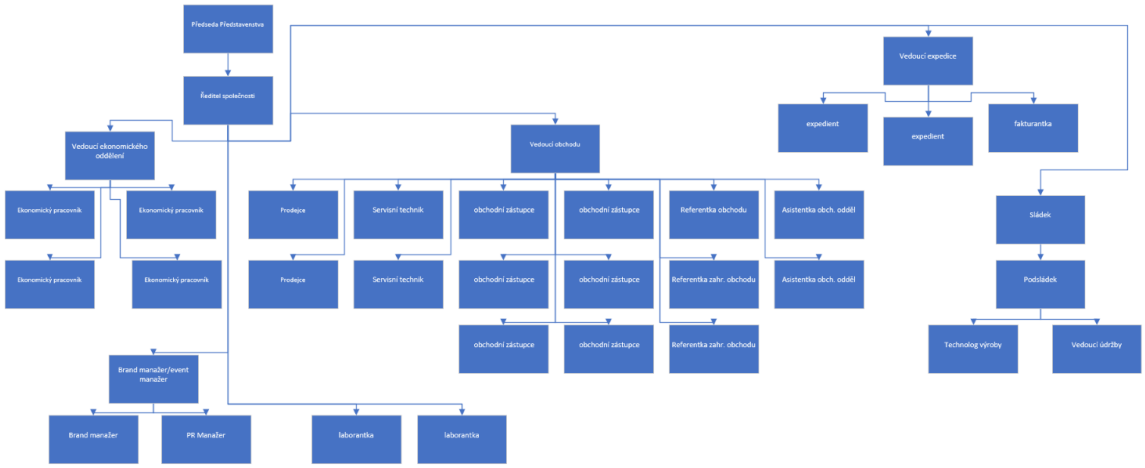
Počet zaměstnanců: 100 - 199 zaměstnanců

Právní forma: Akciová společnost

2.2.Organizační struktura

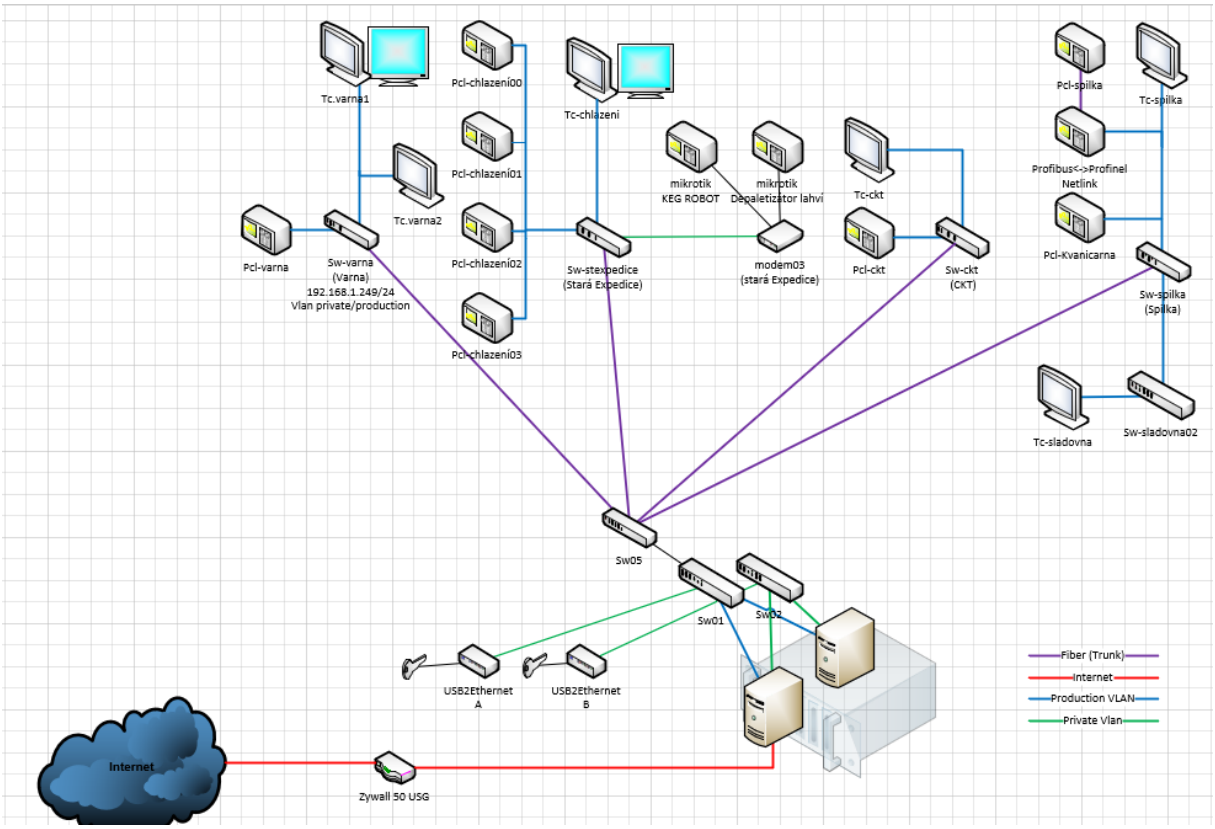
V této firmě funguje Funkcionální organizační struktura, tedy struktura založená na tom, aby byl vždy jiný nadřízený pro různé oblasti fungování organizace. Je zde požadována vysoká odbornost na vedoucích pozicích pro kvalitní chod firmy. Její nevýhodou může být složitost vazeb a komunikace mezi jednotlivými odděleními.

Níže uvidíme model organizační struktury pro tuto firmu. Jsou zde uvedené pouze vyšší pozice, kde jsem nedával např. prodejce v obchodě či řidič vysokozdvižného vozíku.



Obrázek 5 - Organizační struktura zdroj: vlastní zpracování

2.3.Hardwarové vybavení



Obrázek 6 - Síť production zdroj: firemní data

Na výše uvedeném obrázku vidíme vybavení v síti Production. Tato síť zajišťuje komunikaci ve výrobně firmy.

Ve firmě, především administrativní zaměstnanci, pracuji buďto na stolních počítačích nebo laptopech v poměru 24 notebooků a 35 stolních počítačů. Až na výjimky používají techniku od značky DELL. Nejčastěji vyskytovaný počítač je Dell Optiplex 3050 či Dell Lattitude 5580. Ve firmě se nachází zhruba 59 počítačů a notebooků. Na všech zařízení je k dispozici SSD disk, který disponuje oproti starším HDD rychlejší čtením i zápisem. Ve firmě je nainstalováno více než 30 kamer.

V dnešní době je potřeba mít v takové firmě dostatek tiskáren. Zde se nachází přes 20 tiskáren různých značek ale především využívaná je série LaserJet od značky HP.

2.4.Softwarové vybavení

Počítače a notebooky fungují na operačním systému Windows 11 Pro OEM. Tento operační systém zajišťuje plynulý provoz na všech pracovištích a udržuje počítač aktualizovaný. Oproti Windows 11 Enterprise disponuje systém s přídavkem PRO pouze základními funkcemi, ale je to také levnější volba pro podniky. Windows 11 Pro je proto tedy dnes už typickou volbou pro většinu podniků.

Na všech zařízeních jsou dokoupeny kancelářské balíčky Office Standard 2019 které disponují základní výbavou pro většinu kancelářské práce. MS Office standard obsahuje aplikace Excel, Word, Outlook, OneNote a PowerPoint. Oproti MS Office Professional Plus ale nedisponuje aplikacemi Access a Skype.

2.5.Analýza databázové struktury

CIP = Cleaning in place

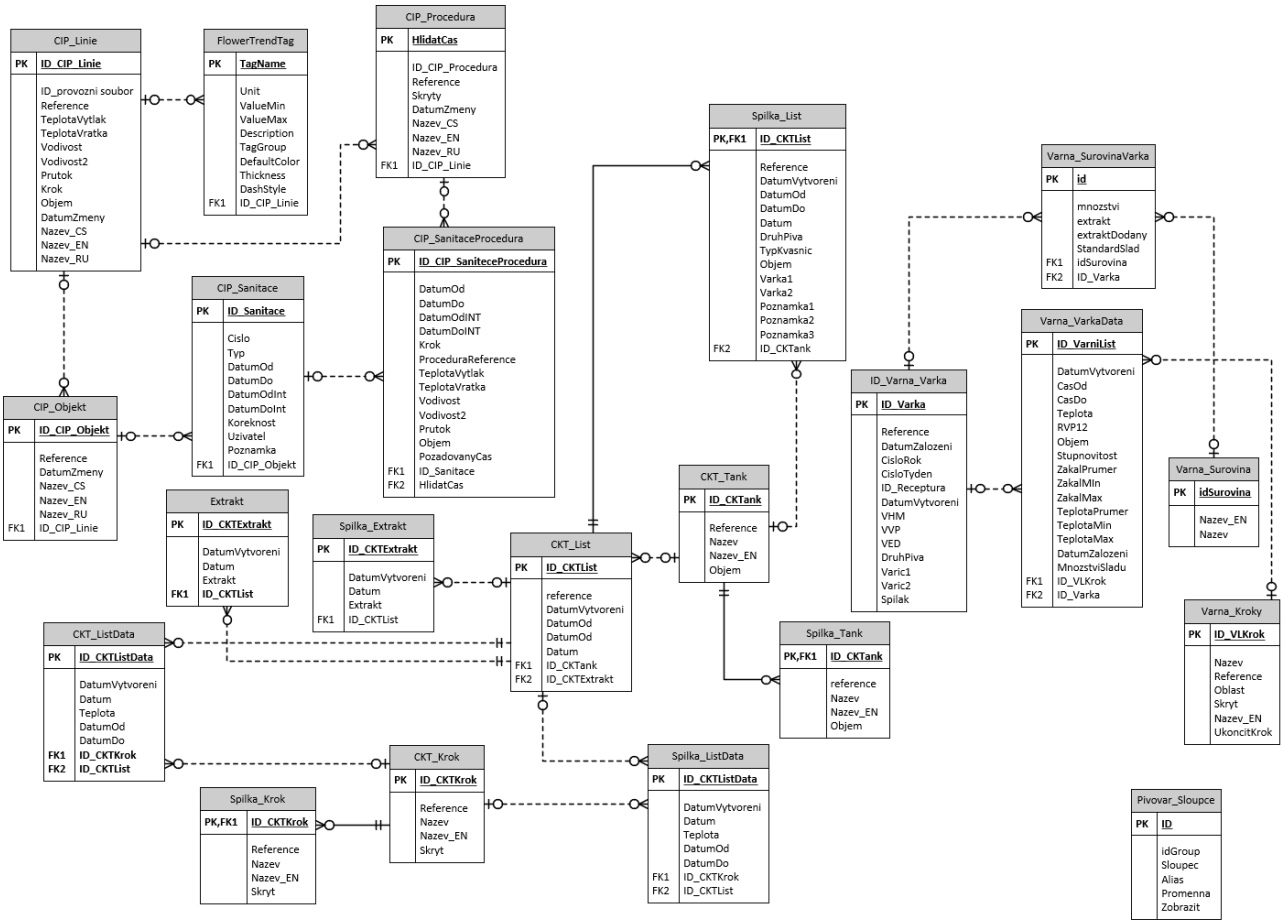
Spilka = Otevřené kádě, kde probíhá kvašení piva.

CKT= Vysokokapacitní uzavřené nádoby používané při kvašení piva.

Databáze této firmy jsou několik let až desítky let staré. K provozu používá rozdělené databáze, ve kterých uchovává data. My si zde popíšeme dvě hlavní databáze, kde uchovávají největší porci dat, a to databázi Production a Databázi FlowerVIS.

Dříve se databáze ukládali odděleně ale dnes už převážně se tabulky vkládají do jedné jediné databáze pro úsporu a jednoduchost dostupnosti dat. Aplikace při ukládání či načítání dat se poté nemusí vícekrát připojovat k databázím, ale stačí jí pouze jedno připojení.

2.5.1. Production

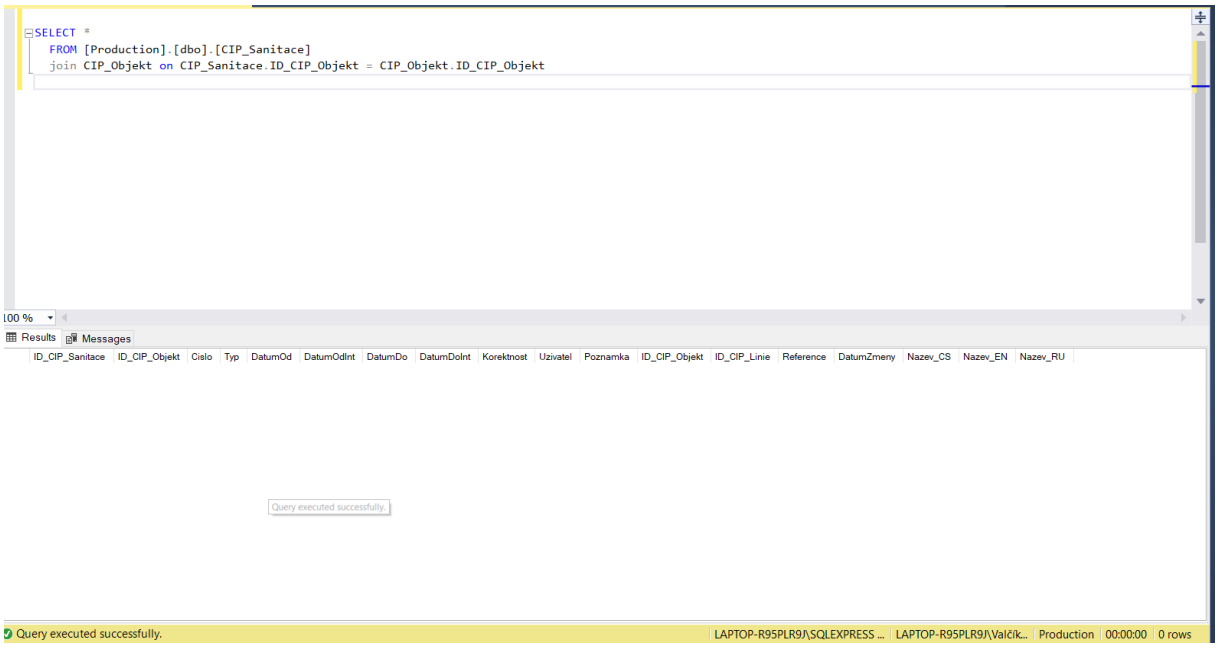


Obrázek 7- Původní struktura Production zdroj: vlastní zpracování

Databáze Production představovala hlavní část ukládání dat z výroby. Dnes je již skoro nevyužívaná. Můžeme uvnitř najít informace o výrobě várek piv a jeho druhu či z jakých surovin se vyrábí. Dále obsahuje informace o Sanitaci (činnosti za účelem odstranění šíření mikroorganismů a škůdců). Informace o využití Spilek, CKT a Tanků.

V této databázi nalezneme velkou spoustu chyb. Tato databáze skrz relace spolu spojuje tabulky, co v reálném prostředí spolu nemají nic společného např. tabulka Spilka_Tank a CKT_Tank. Tyto chyby se zde naskytly několikrát ale z důvodu neidentifikující relace to nemusí dělat problémy a při dotazech které budou správně napojeny, tak může výstupem stále být chtěný výsledek. Ukládání dat z aplikace tedy muselo být přímo do každé tabulky do každého sloupce v řádku, přičemž se aplikace i server zbytečně zatěžuje. Tento návrh je tedy v porovnání s dnešními databázemi velmi neefektivní.

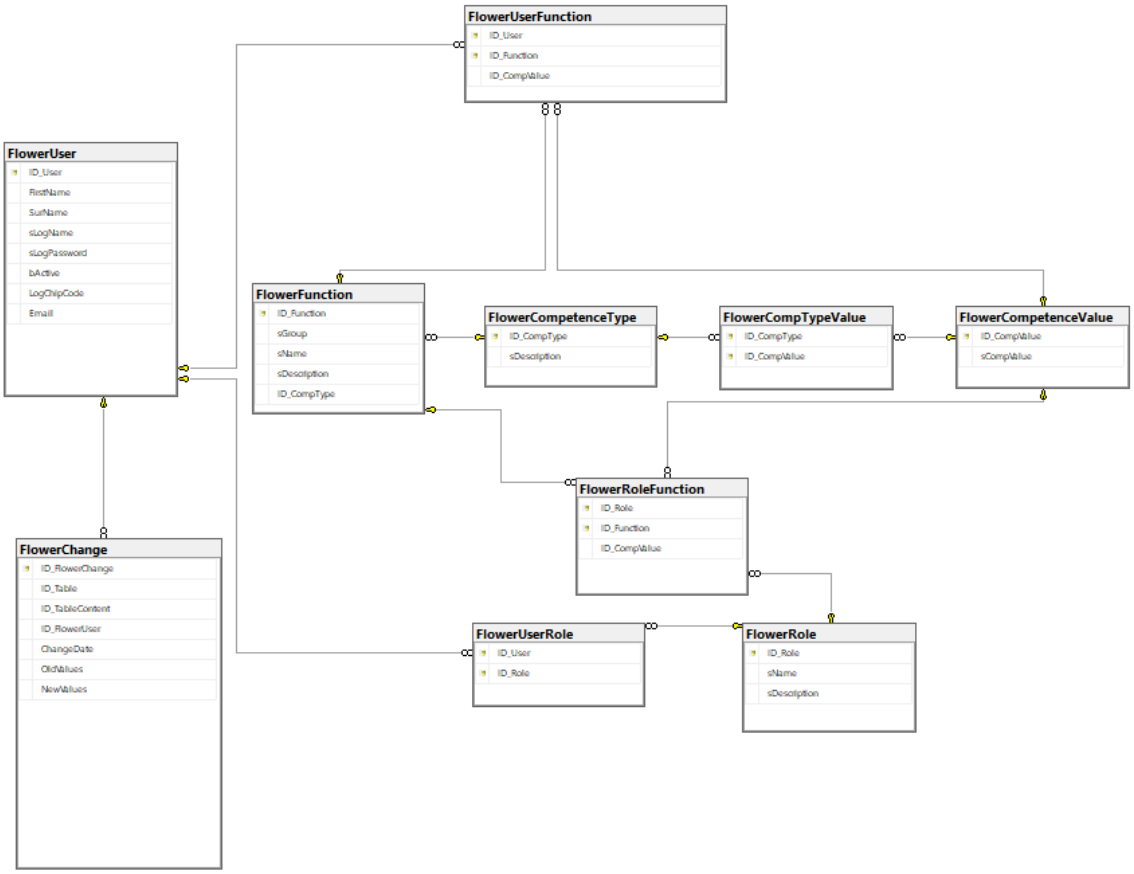
Při průzkumu dat bylo zjištěno, že v tabulkách, co jsou relačně propojeny s CIP_Sanitace nejsou uložena žádná data. Tedy tato část je úplně prázdná tedy by se teoreticky mohla odstranit, pokud se neplánuje používat.



Obrázek 8 - Prázdné tabulky zdroj: vlastní zpracování

2.5.2. FlowerVIS

2.5.2.1. FlowerVIS – USER



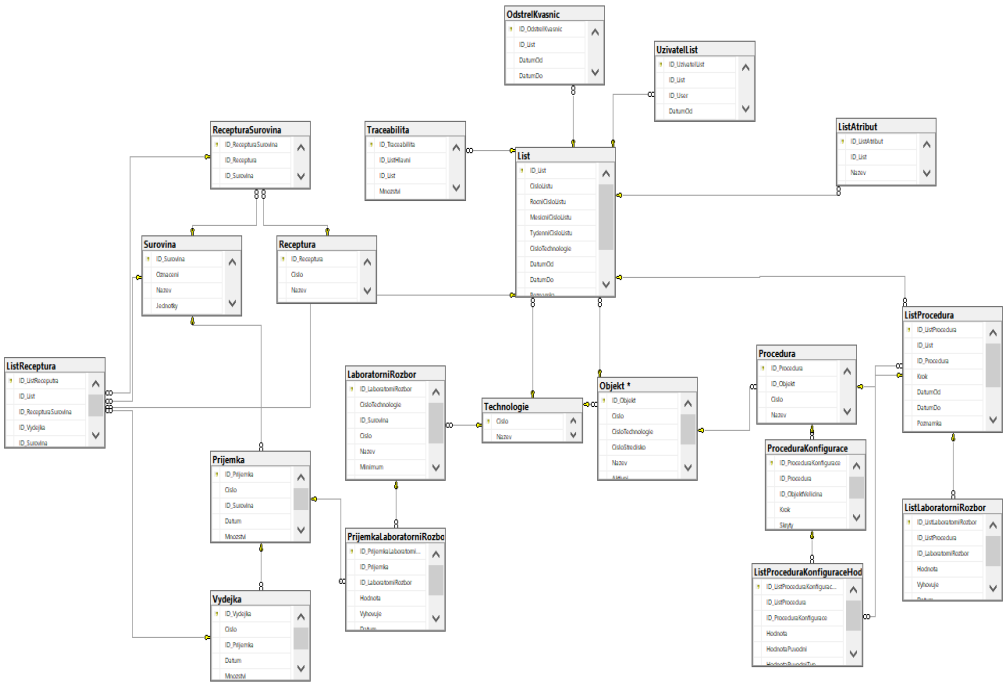
Obrázek 9 - Původní struktura FlowerVIS - User zdroj: vlastní zpracování

Databáze FlowerVIS ukládá pohyb osob. Přesněji při pohybu po areálu pivovaru musí mít každý zaměstnanec čip, se kterým má přístup pouze do některých prostor. Do některých prostor mají přístup pouze vybraní jedinci z důvodu bezpečnosti na pracovišti. Při přístupu do těchto prostor tedy přesněji po „čipnutí“ se uloží záznam u čipu do této databáze. Na výše uvedeném modelu vidíme část databáze, ve kterých jsou uloženy záznamy o personálu, jejich funkci a kam mají přístup.

Zde jsem narazil na některé problematické části. Z Tabulky FlowerUser jde relace do tabulek FlowerUserRole a FlowerFunctionRole které jsou jen propojovací tabulky kvůli relačnímu spojení M:N. Zde tedy bylo použito rozdělení na novou tabulku a dvě oddělená

spojení 1:N. Příliš se mi nelíbila podobnost v tabulkách FlowerRole a FlowerFunction ale z hlediska funkčnosti struktury to takhle může odpovídat realitě.

2.5.2.2. FlowerVIS – Receptury/Rozbor



Obrázek 10 - Původní struktura FloweVIS – Receptury/Rozbor zdroj: vlastní zpracování

Na obrázku č.*** vidíme část databáze FlowerVIS, do které se ukládají receptury k vaření piva, její procedury a laboratorní rozbory. Největší problém těchto databázových struktur se nachází v uložení v od sebe oddělených databázích, které zpomaluje databázový nástroj v práci. Spojením bychom dosáhli lepší optimalizací.

Zde je potřeba se zaměřit na datové typy jednotlivých atributů. Např zde máme atribut název který je datového typu NVARCHAR(255), který mi přijde pro tento účel zbytečně veliký.

Zde můžeme také najít tabulku Objekt, která má v reálném prostředí vazbu i s tabulkou v databázi Production. Proto zde můžeme databáze spojit do jedné velké databáze a docílit požadované optimalizace. Také zde máme atributy ID_USER který se nachází v jiné databázové struktuře. Přesněji ve FlowerVIS – USER proto tedy můžeme propojit i s touto strukturou.

3. Vlastní návrh řešení

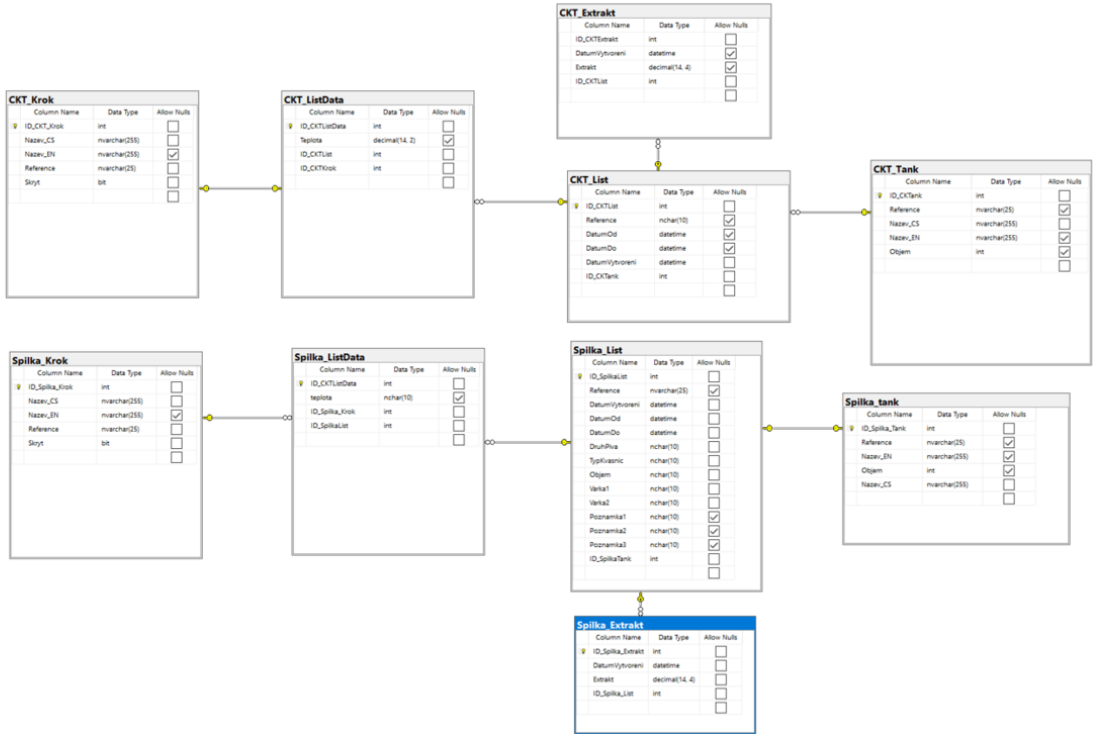
V následující kapitole představím hlavní část této bakalářské práce. Budu se zabývat řešením problému databázové struktury, a proto je důležité si znovu připomenout chyby a nedostatky, které v těchto strukturách nastali.

Jako největší problém zde je tedy databázová struktura v databázi Production. Tabulky jsou špatně spojené relaci, které v reálném prostředí neexistují tedy je potřeba je upravit pro optimalizaci této struktury.

Dalším problémem, na který jsem při analýze narazil, bylo tedy rozdělení databázových struktur do samostatných databází, které zde není potřeba. Proto se budu zabývat sloučením těchto struktur do jedné velké databáze za účelem optimalizace a zrychlení práce aplikace s databázovým nástrojem.

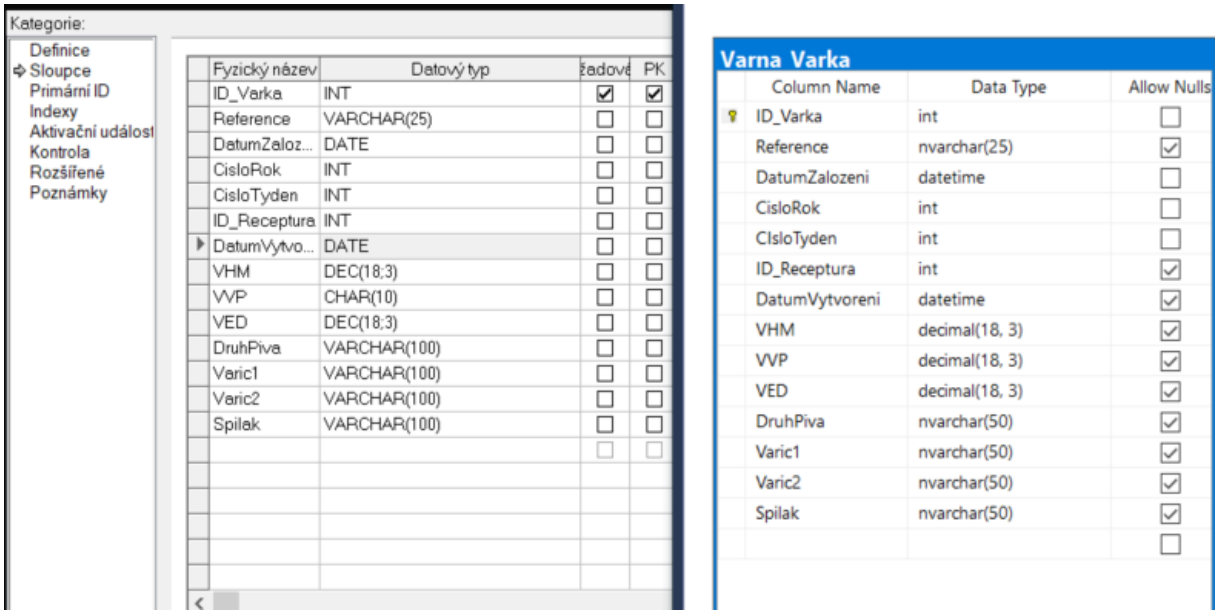
3.1.Řešení databáze Production

Zde jsem se musel pořádně podívat na data v tabulkách a popřemýšlet o reálných vztazích. Některé relace zde byly neopodstatněné, proto jsem je musel upravit, nebo dokonce odstranit. Na obrázku. Č.7 jsme viděli jak tabulka CKT_List je relačně spojena s tabulkou CKT_ListData a úplně stejně s tabulkou Spilka_ListData, jenže druhá zmiňovaná tabulka nemá s CKT_ListData nic společného. Stejně jako CKT_Krok s tabulkou Spilka_Krok. Proto jsem tyto tabulky od sebe oddělil a řešení vypadalo poté tímto stylem



Obrázek 11 - Řešení databáze Production zdroj: vlastní zpracování

Strukturu u Varna_Varna jsem nechal být. Zde nebyla porušena žádná normální forma, proto tedy ponechávám tuto část bez svého výraznějšího zásahu do struktury databáze. Zde jsem odstranil prázdnou tabulku Pivovar_Sloupce. A poté jsem upravil datové typy, které lépe odpovídají reálnému prostředí.



Obrázek 12 - Nastavení datových typů zdroj: vlastní zpracování

Jako příklad jsem si vybral tabulku Varna_Varka. Na obrázku tedy vidíme na levé straně starý model s datovými typy a příznaky „požadovat hodnoty“ a „PK“. To znamená že pokud je zakštnuté „požadovat hodnoty“, tak do těchto atributů při vkládání dat se vždy musí uložit data. Tedy nelze nelze uložit do této tabulky řádek bez této informace. PK je tedy primární klíč.

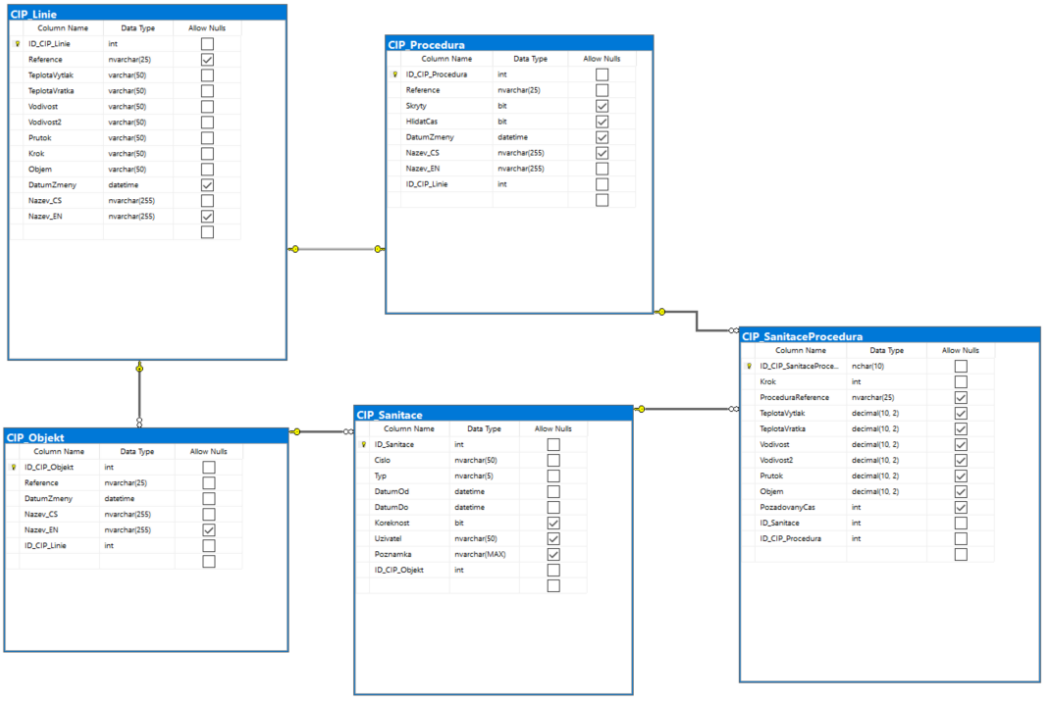
Vidíme zde že u datumů byl použit datový typ DATE tedy ve formátu (RRRR-MM-DD) což je dle mého názoru nedostatečné a potřebujeme zde ukládat i bližší informace o čase. Proto jsem tedy použil datový typ DATETIME (RRRR-MM-DD HH:MM:SS:[NNN]) který ukládá hodiny i minuty atd.

Dále jsem pozměnil datový typ u znaků z VARCHAR na NVARCHAR. Typ VARCHAR je omezen na 8bitový kód, přičemž NVARCHAR může uložit jakékoliv Unicode data. Tedy NVARCHAR. Pokud tedy chceme zamezit možným

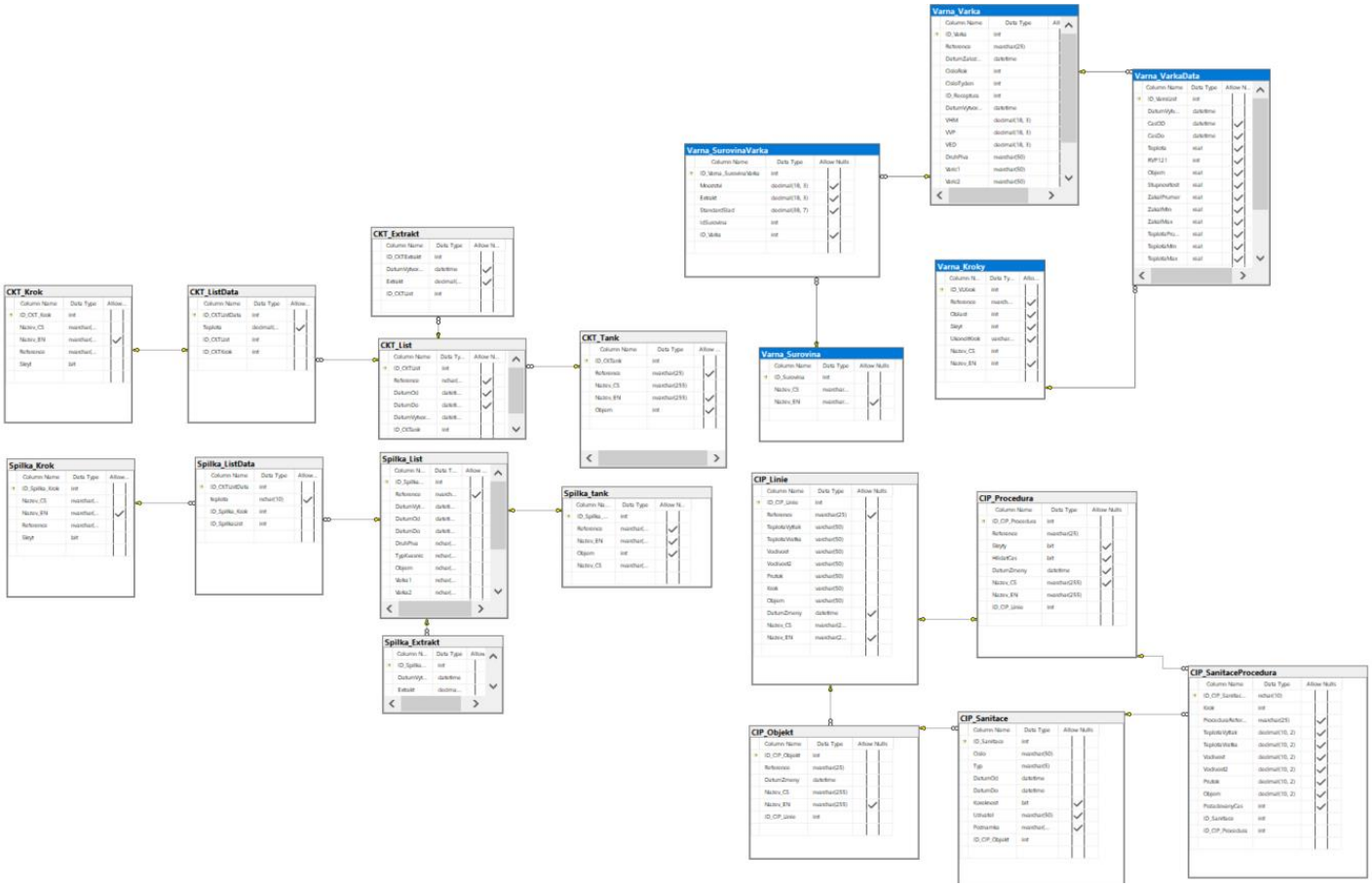
problémem s kompatibilitou použijeme NVARCHAR. Nevýhodou NVARCHAR je že využívá více místa. Také jsem snížil počet znaků co do nich můžeme vložit za účelem úspory prostoru. V datech jsem nenašel více jak 30 použitých znaků proto tedy 50 znaků by mělo stačit.

Zajímavostí bylo, že dříve byl použit v atributu VVP datový typ CHAR(10) přestože zde byly uloženy typově stejná data jako ve VHM. Proto se tedy číselná data ukládala jako text a mohlo to dále vykazovat problémy při manipulaci s těmito daty. Proto jsem tedy při úpravě použil datový typ DECIMAL(18, 3)

U některých atributů jsem nastavil příznak NotNull. U těchto atributů mi přišlo podstatné, aby tyto informace při ukládání v sebe měli data a aby bez nich záznam nemohl existovat tedy u atributu DatumZalozeni, CisloRok a CisloTyden.



Obrázek 13 Řešení tabulek CIP zdroj: vlastní zpracování



Obrázek 14 - Úplné řešení Production zdroj: vlastní zpracování

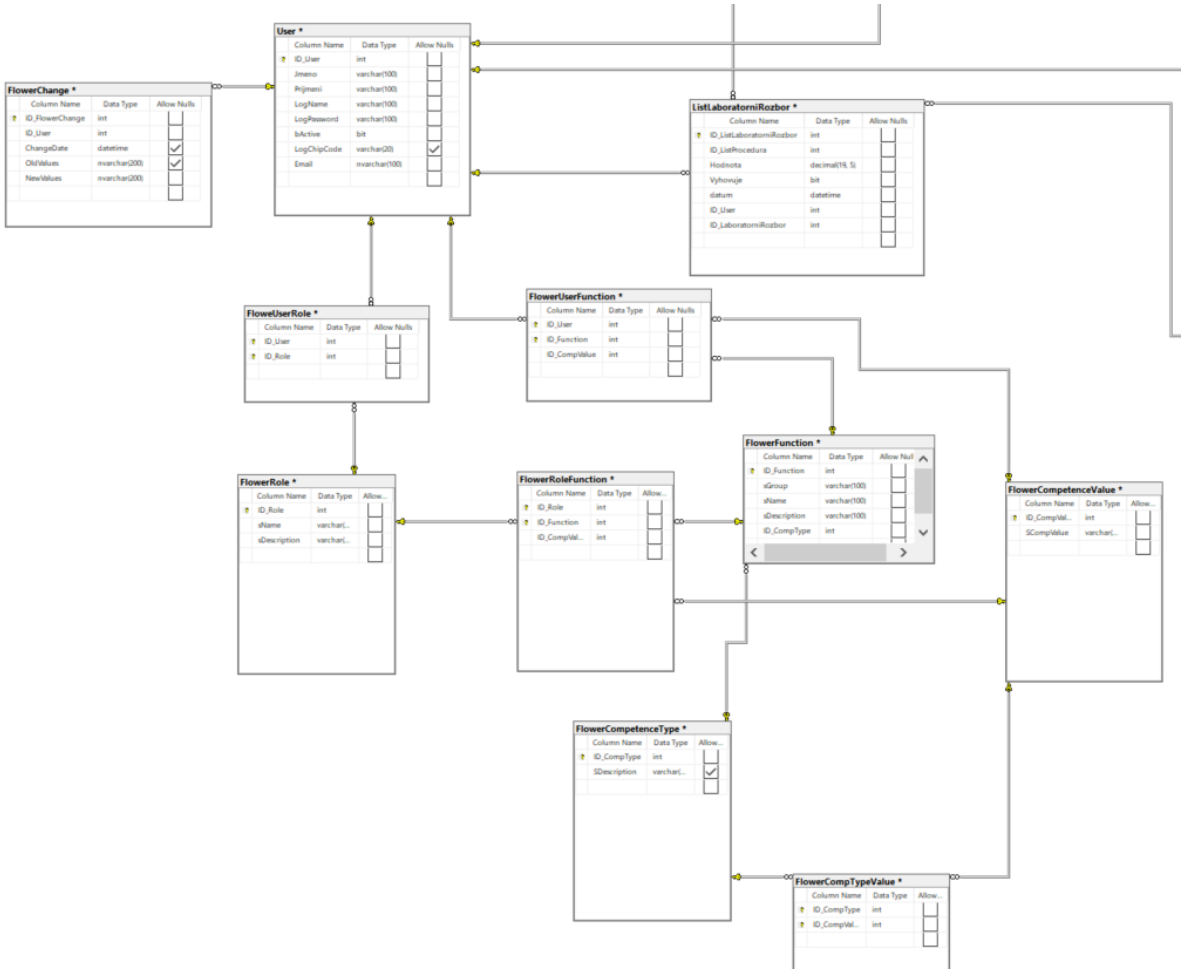
Při úpravě zbylé části, což je databázová struktura s tabulkou CIP_Sanitace. Zde sem postupoval nejdříve kontrolou dat uvnitř tabulek. V tabulce FlowerTrendTag jsem nenašel žádné data a žádné bližší spojení s tabulkami tak jsem jí odstranil ze struktury. Dále jsem oddělal za mne třetí nedůležitý název v Ruském jazyce. Dále jsem upravil datové typy obdobně jak u tabulky Varna_Varka.

3.2. Řešení databáze FlowerVIS

3.2.1. FlowerVIS – User

Zde po analýze jsme zjistili, že tu jsou v reálném prostředí problematické relace M:N a proto tu byly použity spojovací tabulky, které tuto vazbu rozdělují na 1:N. Poté zde je k vidění oddělené tabulky FlowerUserRole a FlowerUserFunction. Zde jsou tyto tabulky odděleny z důvodu, že v reálném prostředí, když si určíme třeba dvě různé role např. administrátor a obyčejný uživatel, tak administrátor má funkci prohlížení, vkládání dat či úpravy, přičemž uživatel bude mít třeba jen funkci prohlížení. Proto musí být tyto tabulky oddělené protože různé role můžou mít různé funkce tedy znovu vazba M:N.

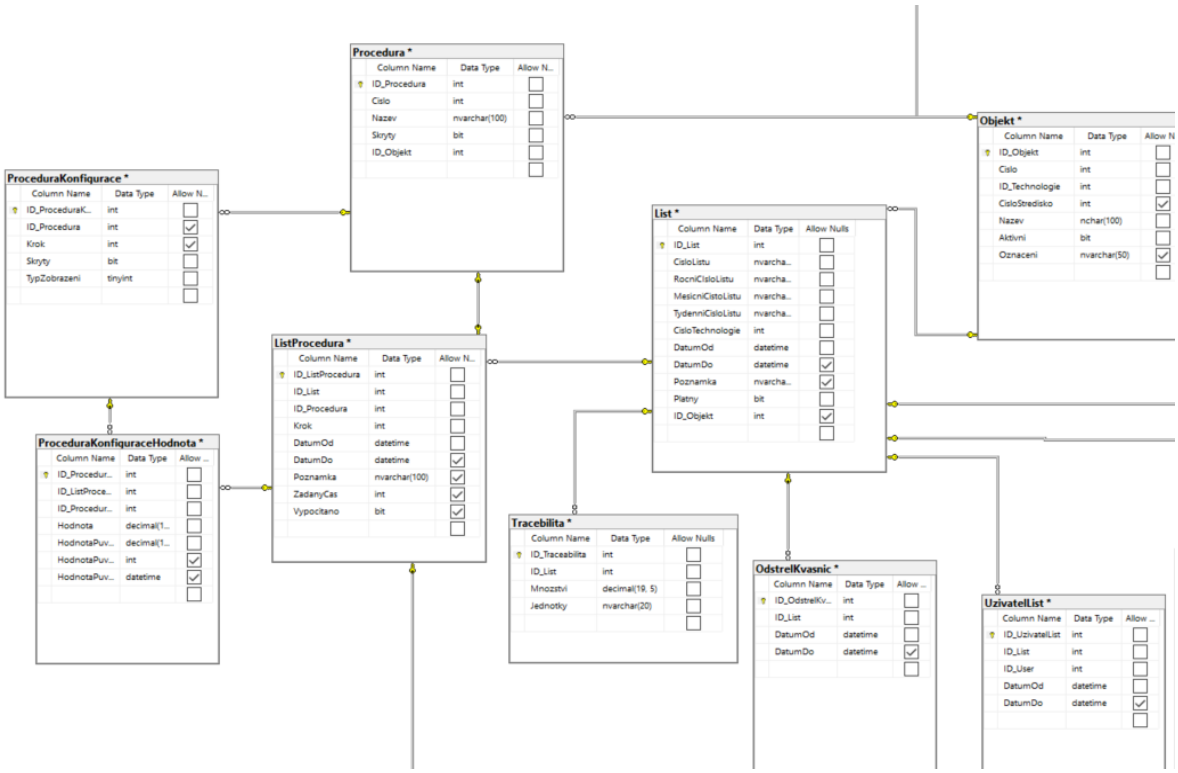
Poté tedy po malých úpravách datových typů a zda hodnoty můžou být NULL vypadá struktura databáze FlowerVIS – User takto:



Obrázek 15 - 1. část upravené Databáze FlowerVIS - User zdroj: vlastní zpracování

3.2.2. FlowerVIS – Receptury/Rozbor

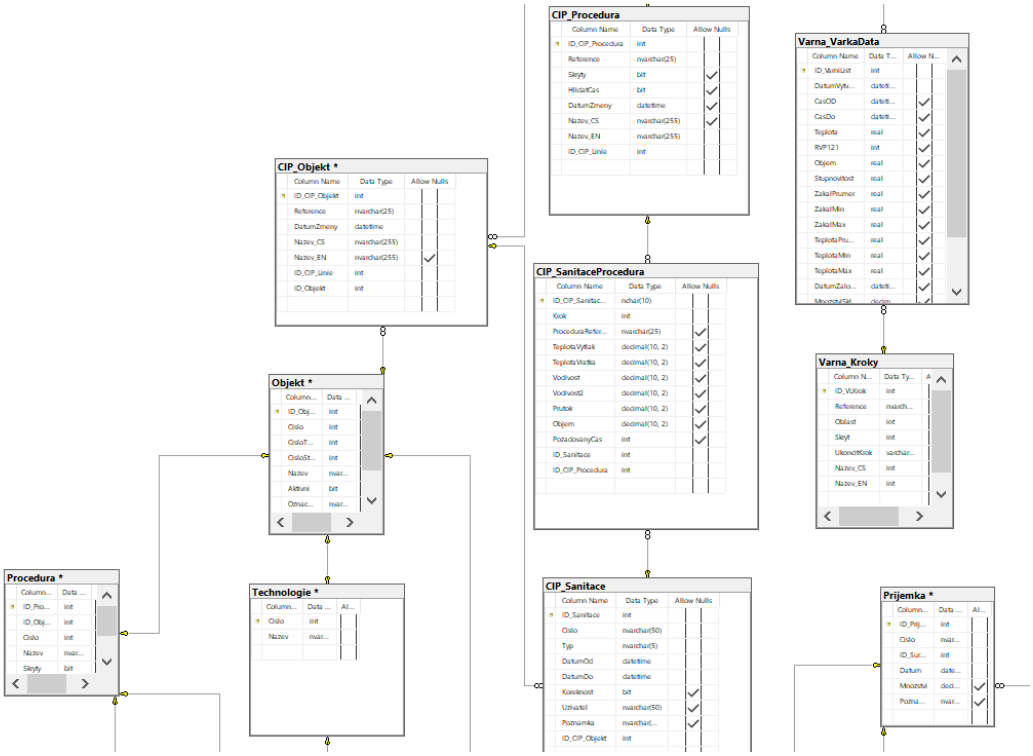
Zde po analýze jsem musel vyřešit problém s datovými typy. Bylo zde mnoho tabulek a mnoho chyb v určování jednotlivých typů atributů. U většiny případů zde byl využit samostatný CHAR který oproti typu VARCHAR či NVARCHAR je fixní a pokud zadáme např. CHAR(100) tak poté data využívají vždy plnou délku i když jsou data menší. Pokud tedy nevyužijeme celou délku, mají zbylé znaky nulovou hodnotu oproti tomu VARCHAR je variabilní a určuje se aktuální délku v paměti podle počtu zadaných znaků.



Obrázek 16 - 2. část upravené Databáze FlowerVIS – Procedury zdroj: vlastní zpracování

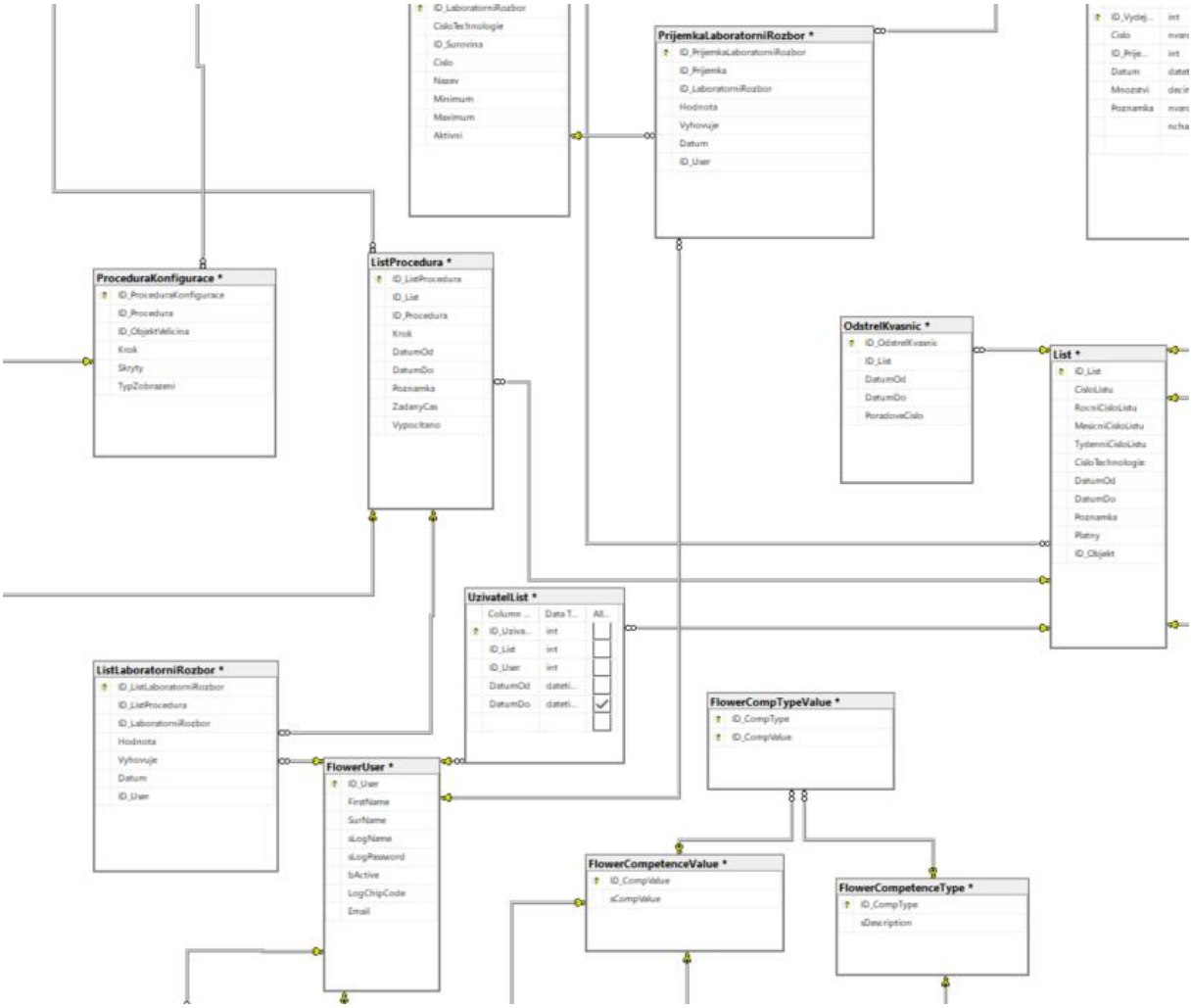
3.3.Řešení spojení databází

Jelikož jak bylo výše zmíněno, z hlediska dostupnosti a ulehčení přístupu je výhodnější mít všechny data a tabulky v jedné databázi. Proto jsem zde využil k propojení a snadnějšímu čtení dat tabulky Objekt a CIP_Objekt kde v tabulkách byly ukládána zhruba stejná data. Proto jsem zanechal obě tabulky a přidal pouze do tabulky CIP_Objekt atribut ID_Objekt s relací.



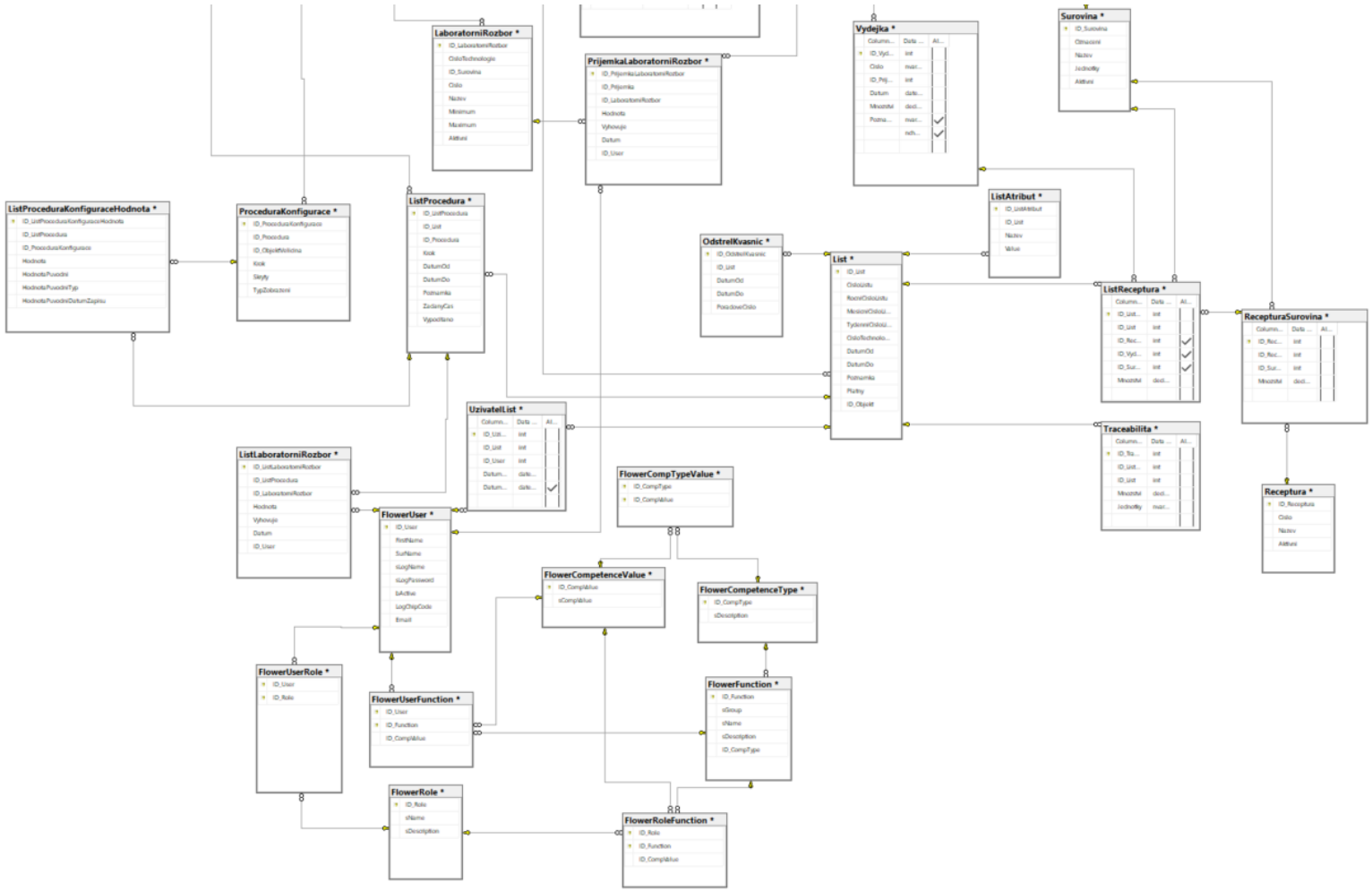
Obrázek 17 - Řešení spojení databáze s CIP zdroj: vlastní zpracování

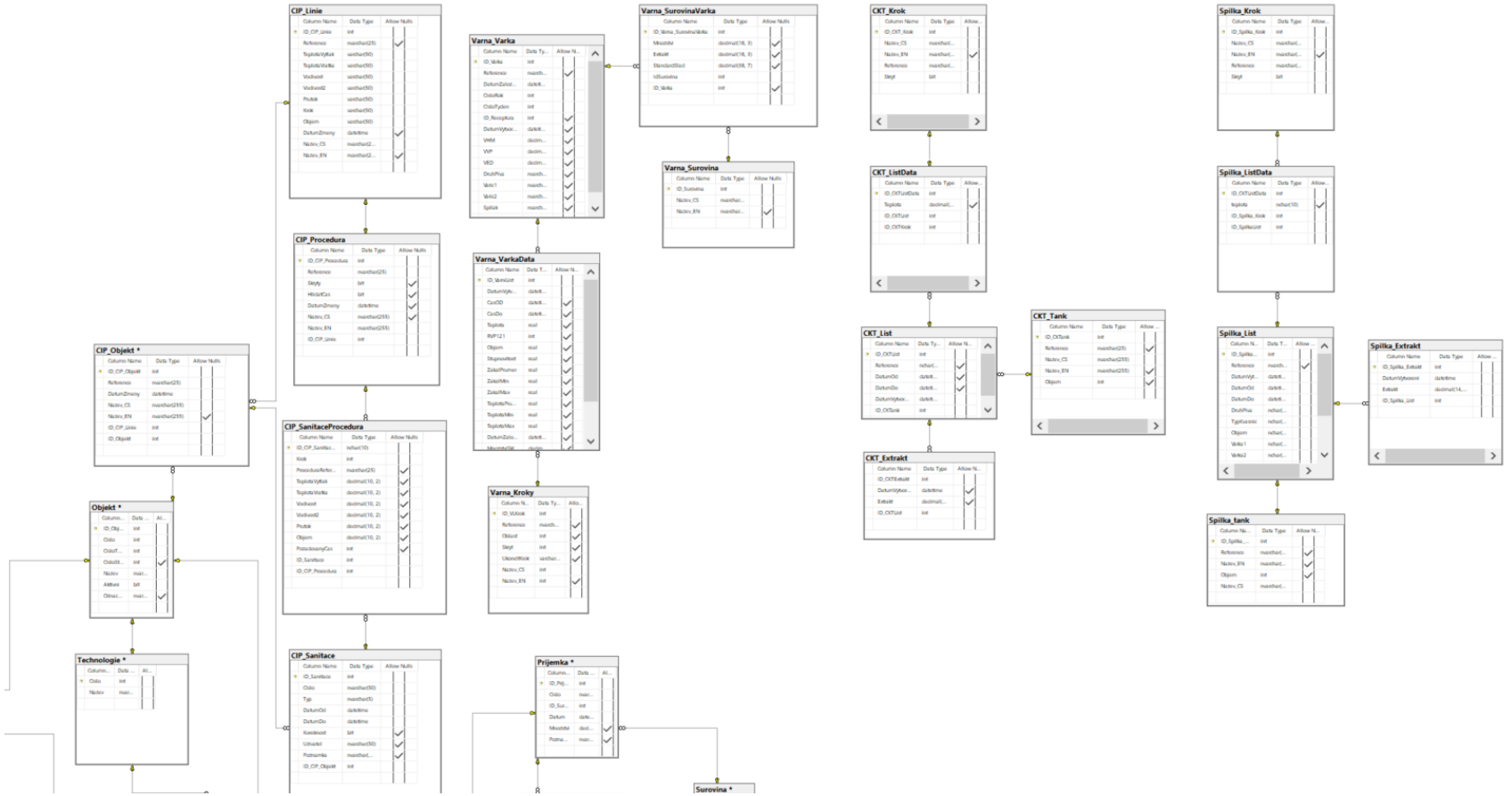
Pro propojení tabulek s FlowerVIS – USER jsem použil tabulku FlowerUser kde v některých tabulkách jako je například PrijemkaLaboratorniRozbork která už v sobě měla atribut ID_User ale z mne neznámého důvodu nebyla propojena. Proto následující obrázek popisuje moje řešení propojení s FlowerVIS – USER



Obrázek 18 – Řešení spojení dataváze s FlowerVIS zdroj: vlastní zpracování

Poté by tedy celá databáze mohla vypadat takto kompletně spojená.





Obrázek 19 - Celková upravená Databáze zdroj: vlastní zpracování

3.4. Předpokládané přínosy pro podnik

Hlavním cílem této bakalářské práce byla analýza a úprava modelu databáze pro pivovar, ve kterým je potřeba zaznamenávat informace o výrobě piva, o uživatelích, jejich rolích a o pohybu osob po pracovišti pomocí čipů. Model by měl být úspornější v zabraném místě na serveru. kde je tato databáze uložena. Je potřeba v aplikačním rozhraní při implantaci tohoto modelu zasáhnout do ukládání dat. Je pravděpodobné, že v ojedinělých případech zde nemusí nově upravený model odpovídat aktuálně ukládaným datům, a proto je potřeba je v aplikaci upravit nastavení.

Přínosy pro firmu jsou zejména v administrativní části podniku, tedy bude jednodušší spravování rolí a práv uživatelů. Díky novému modelu databáze bude možné mnohem rychleji a účinněji získávat údaj z databáze. Díky úpravě datových typů jsem dosáhl lepší konzistence dat.

Využití tohoto modelu přináší výhody v přístupnosti dat, tedy všechny data jsou uložena na jednom místě a není potřeba se vícekrát připojovat do různých databází, které ale běží na jednom serveru.

4. Závěr

Hlavním cílem mé bakalářské práce bylo zajištění optimalizace databázové struktury podniku s činností v oboru pivovarnictví s názvem XXX Beer Brewery a.s.

Při prvotní analýze struktury databáze jsem nedokázal najít výraznější chyby ve struktuře, nepočítaje menší chyby v datových typech. Až po rozhovorech s kolegy jsem objevil pár výraznějších prohřešků. Nejobtížnější pro mne bylo rozpoznat důležitost některých entit a jejich atributů, jelikož v datech, které mi byly dodány, nebyly žádné záznamy. Proto bylo složité rozhodování, zda je ponechat či nikoli.

Výsledkem této bakalářské práce je tedy to, že na základě dodaných dat z podniku, jsem byl schopen analyzovat aktuální stav databázové struktury, a k tomu navrhnout patřičné řešení. Fyzický návrh jsem vytvářel na Microsoft SQL Server v programu MS Management Studio. Cíl práce byl tedy splněn.

Toto téma pro mě bylo velmi složité. Od počátku jsem věděl že podnik je rozsáhlý a už má zaběhlou funkční databázi pro jejich činnost, což znamená velká porce dat a velká struktura kterou bude potřeba analyzovat. Díky čemuž jsem byl donucen si rozšířit své znalosti v oboru databázových modelů a jejich implementací.

Literární prameny

[1] KOCH, M., NEUWIRTH, B. Datové a funkční modelování. 3. vyd. Brno: Akademicke nakladatelství CERM, 2008. 121 s. ISBN 978-802-1437-319

[2] Skřivan J. Datové modely a návrh relačních schémat.Praha: UK Filozofická fakulta 2008 [online] Dostupné z: <https://sites.ff.cuni.cz/uisk/wp-content/uploads/sites/62/2016/01/Datov%C3%A9-modely-a-n%C3%A1vrhy-rela%C4%8Dn%C3%ADch-sch%C3%A9mat_Sk%C5%99ivan.pdf>

[3] OTTE L. Databázové systémy Ostrava: VSB Fakulta strojní. 2013 [online] Dostupné z: https://projekty.fs.vsb.cz/463/edubase/VY_01_044/Datab%C3%A1zov%C3%A9%20syst%C3%A9my.pdf

[4] JAŠEK, Petr. E-learningový kurs pro výuku jazyka SQL. Brno, 2008. [online]. Dostupné z: http://hdl.handle.net/11012/52950. Bakalářská práce. Vysoké učení technické v Brně. Fakulta informačních technologií. Ústav informačních systémů. Vedoucí práce Šárka Květoňová.

[5] ŘÍHA, Petr. Návrh SQL databáze pro podporu činností malé IT firmy [online]. Brno, 2012. Dostupné z: http://hdl.handle.net/11012/7828. Bakalářská práce. Vysoké učení technické v Brně. Fakulta podnikatelská. Ústav informatiky. Vedoucí práce Jiří Kříž.

[6] HERNANDEZ, Michael J. Návrh databází. 1.vyd. Praha: Grada, 2006. 408s. ISBN 80-247-0900-7.

[7] POKORNÝ, J. a HALAŠKA, I. Databázové systémy. 2.vyd. Praha: ČVUT, 2003.148s. ISBN 80-01-02789-9.

[8] SHELDON, Robert. SQL : začínáme programovat [online]. Praha : Grada, 2005 [cit. 2023-05-11]. ISBN 80-247-0999-6. Dostupné z: https://primo.lib.vutbr.cz/permalink/f/1roshr/420BUT_Aleph000117271

[9] Madron L. DATOVÉ SKLADY A OLAP V PROSTŘEDÍ MS SQL SERVERU. Brno: VUT Fakulta informačních technologií [online]. Dostupné z: https://www.vut.cz/www_base/zav_prace_soubor_verejne.php?file_id=116351

[10] What is SQL Server Management Studio (SSMS). In: Microsoft [online]. 2023 [cit. 2023-05-08]. Dostupné z: https://learn.microsoft.com/en-us/sql/ssms/sql-server-management-studio-ssms?view=sql-server-ver16

Seznam obrázku