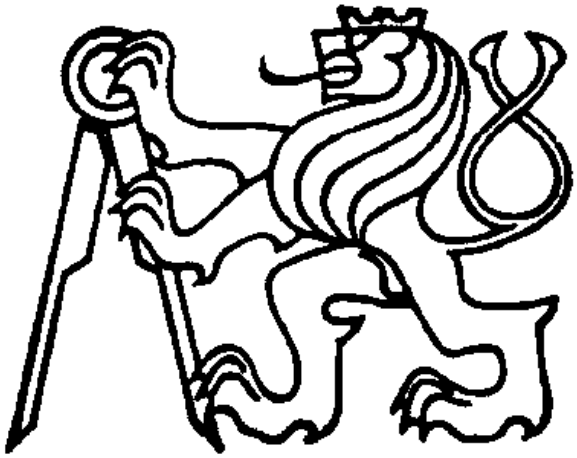
ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE

FAKULTA STAVEBNÍ

Katedra ekonomiky a řízení ve stavebnictví



DIPLOMOVÁ PRÁCE

2018 Bohdan Povýšil

ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE
Fakulta stavební
Thákurova 7, 166 29 Praha 6

ZADÁNÍ DIPLOMOVÉ PRÁCE

I. OSOBNÍ A STUDIJNÍ ÚDAJE

Příjmení: Povýšil Jméno: Bohdan Osobní číslo: 380607

Zadávající katedra: Katedra ekonomiky a řízení ve stavebnictví

Studijní program: Stavební inženýrství

Studijní obor: Projektový management a inženýring

II. ÚDAJE K DIPLOMOVÉ PRÁCI

Název diplomové práce: Evidence majetku ve stavební společnosti

Název diplomové práce anglicky: Property evidence in construction company

Pokyny pro vypracování:
Úvod do problematiky evidování majetku
Zavedení metodiky evidence
Výpočetní a databázové systémy vhodné pro evidenci
Práce s danou evidencí v rámci stavebního podniku

Seznam doporučené literatury:
MOLNÁR, Zdeněk. Podnikové informační systémy. Vyd. 2., přeprac. V Praze: České vysoké učení technické, 2009, 195 s. ISBN 978-80-01-04380-6.
Project Management Institute. A Guide to the Project Management Body of Knowledge (PMBOK Guide). Newtown Square, Pa: Project Management Institute, 2004.
Amy Ditmar: Corporate Financial Analysis, McGraw Hill, 2000

Jméno vedoucího diplomové práce: Ing. Radan Tomek, MSc.

Datum zadání diplomové práce: 28.6.2017 Termín odevzdání diplomové práce: 7.1.2018
Údaj uveďte v souladu s datem v časovém plánu příslušného ak. roku

Podpis vedoucího práce Podpis vedoucího katedry

III. PŘEVZETÍ ZADÁNÍ

Beru na vědomí, že jsem povinen vypracovat diplomovou práci samostatně, bez cizí pomoci, s výjimkou poskytnutých konzultací. Seznam použité literatury, jiných pramenů a jmen konzultantů je nutné uvést v diplomové práci a při citování postupovat v souladu s metodickou příručkou ČVUT „Jak psát vysokoškolské závěrečné práce“ a metodickým pokynem ČVUT „O dodržování etických principů při přípravě vysokoškolských závěrečných prací“.

Datum převzetí zadání Podpis studenta(ky)

Anotace

Práce se zabývá problematikou evidence majetku ve stavební společnosti. Autor práce popisuje specifika práce s majetkem ve stavební společnosti a má za cíl vytvořit model evidence v databázovém softwaru. Práce obsahuje teoretický popis vývoje návrhu, který se dělí na konceptuální, logický a fyzický návrh. Na ten navazuje popis samotných specifik evidence v stavebním podniku. Inspirací od podobných modelů se vytvořil model databáze. Metodou porovnání autor vybral software, ve kterém databázi podle modelu vytvořil, a navrhl metody pro urychlení práce s databází.

Annotation

The work covers the problematics of evidence of assets in construction company. The author describes the specifics of the work with assets in construction company and has a goal to create a model of an evidence by using database software. The work contains a teoretical description of the design development, which is divided to the conceptual, logical and physical part. This is followed by a specifics description of construction company evidence. The model was created by inspiration from similar models. The author used comparison method to choose software he used to create database according to the model, and he suggested methods to speed-up the proces of work with database.

.

Klíčová slova:

Evidence, majetek, databáze, stavebnictví, stavební podnik

Key words

Evidence, asset, property, database, civil engineering, construction company

…

Obsah

1 Úvod

Stavebnictví je obor s množstvím technologických procesů využívající mechanizaci, která usnadňuje, urychluje nebo vůbec umožňuje realizaci stavebních projektů. Tato mechanizace může být různého charakteru; může být jednoúčelová pro jeden daný projekt, nebo naopak se může jednat o stroj, se kterým se počítá na realizaci vícero projektů. Výhodnost koupě mechanizace je úzce spojená s jejím efektivním využitím. Investice do mechanizace tedy může být velmi výhodná, ale zároveň se může stát zátěží podniku, není-li využíván její potenciál. Navíc zejména u větších podniků může nastat situace, že se na daný majetek, mechanizaci, tedy předchozí investici do ní, zapomene, obzvlášť, pokud neexistuje potřebná evidence tohoto majetku.

Evidence je ve stavebnictví, podobně jako v jiných odvětvích, důležitým faktorem pro úspěšné hospodaření podniku. Zamezuje ztrátám, lze z ní zjistit hodnotu podniku, udává informace o stavu, momentální lokalitě a jiných informacích o majetku. Jedná se o soustavné vedení záznamů, které je v dnešní době nedílnou součástí podnikání, nejen z hlediska povinného účetního vedení záznamů, ale i konkurence, která evidenci využívá ke zefektivnění řízení v rámci podniku. Ku příkladu už Tomáš Baťa zavedl evidenci o výrobcích a zákaznících, kterou poté využil nejen při propagování svého produktu, ale také ke zlepšení kvality svých výrobků tím, že tyto evidované zákazníky jeho zaměstnanci obcházeli a sbírali jejich připomínky.

Ve stavebnictví v souvislosti s evidencí jde zejména o získání přehledu nad majetkem podniku, který je v oběhu, který něco vytváří, který pro podnik vydělává peníze. Je vhodné mít přehled o majetku, o jeho využití a ideálně i plán, kdy ho na daný projekt přiřadit. Největším ztrátám totiž ve stavebnictví dochází při neefektivním procesu přípravy a realizace projektů. Možnost mít evidenci, kde je možné mít okamžitý přehled o svých podnikových možnostech může mít velký vliv na rozhodování či výběru projektů, které chce podnik realizovat. Dnešní propojená doba formou datových sítí jen napomáhá k vedení a využití evidence pomocí databázových systémů, které jsou již dostupné v čase kdekoliv v dosahu internetového připojení.

Tato práce má za cíl navrhnout základy takového funkčního systému.

2 Evidence a její problematika

Pro začátek si musíme uvědomit, co vlastně chceme. Evidence majetku je vlastně databází spravovaného majetku s daty, které s ním souvisí. Jedná se tedy o problém, který se řeší ve spoustě odvětvích včetně stavebnictví. Cílem práce je vytvořit **databázi**, která by byla podkladem pro vedení evidence majetku ve stavební společnosti. Pro řešení tohoto problému je vhodné se seznámit s historií a metodikou tvorby a správy těchto databázových systémů, což je také obsahem této kapitoly.

2.1 Historie databázových systémů

V historii byla evidence řešena pomocí kartoték, do kterých se zapisovaly položky, rozřazovali se podle kategorií, evidovali se jejich změny a případně se řešili případné nesrovnalosti. Celá práce s kartotékami a evidencí byla řešena ručně. Celý tento proces byl logicky zdlouhavý a oproti dnešním požadavkům neefektivní, v některých podnicích se ovšem používá dodnes.

Dalším krokem ve vývoji zpracování dat bylo převedení tohoto procesu na stroje. Prvním takovým příkladem bylo sčítání lidu v roce 1890 ve Spojených státech Amerických. Během sčítání lidu v tomto roce byly využity stroje, které využívali děrné štítky jako paměťové médium. Data z těchto štítků byla poté zpracována pomocí těchto strojů. Tento druh zpracování dat byl využíván po celém světě po další půl století.

Jak jsem již zmínil, stroje využívali (a někdy stále využívají) pro zápis dat děrné štítky. Jsou vyráběny z tenkého kartonu a informace se zapisuje formou děrování na určité pozici. Místa pro otvory jsou uspořádaná do matice. Na těchto štítcích bylo umístěno až 90 sloupců pro záznam dat. Jeden štítek obsahoval data o velikosti 960 bitů, což je 120 bajtů. Vzhledem k objemu dat, které se obyčejně využívají dnes, je využití děrných štítků dávno překonané. Je ovšem vhodné sdělit, že se děrné štítky využívali při vývoji jaderné bomby v projektu Manhattan, ve kterém pomáhali řešit výpočty průběhu reakcí.

Velkou změnou při vývoji databází bylo vyvinutí magnetické pásky pro účely uchování dat společností Univac v roce 1950, a následným vyvinutím prvního komerčního počítače o rok později, takzvaného UNIVAC I. I přesto, že tento počítač byl vyroben jen v 46 kusech kvůli ke své vysoké ceně (až 1 500 000 $), nastartoval vývoj počítačové technologie a správy dat. S tímto vývojem počítačů obecně se

ukázalo, že původní univerzální používání strojového kódu procesorů je nejen pro databáze neefektivní. S tímto zjištěním se objevil požadavek na vývoj vyššího jazyka pro zpracování dat (COBOL). Vyvinuly se i efektivnější způsoby zpracování dat, jakým je například dávkové zpracování. S postupným vývojem se objevil požadavek na vytvoření online databází, a s tímto požadavkem se musela definovat koncepce databázových systémů. To se stalo na konferenci CODASYL v roce 1965. V roce 1971 byla vydána publikace The DBTG April 1971 Report, ve kterém se poprvé objevily pojmy jako schéma databáze, subschéma, jazyk pro definici schématu a jiné, a byla definována celá architektura síťového databázového systému.

2.2 Pravidla tvorby databází a jednotlivé modely

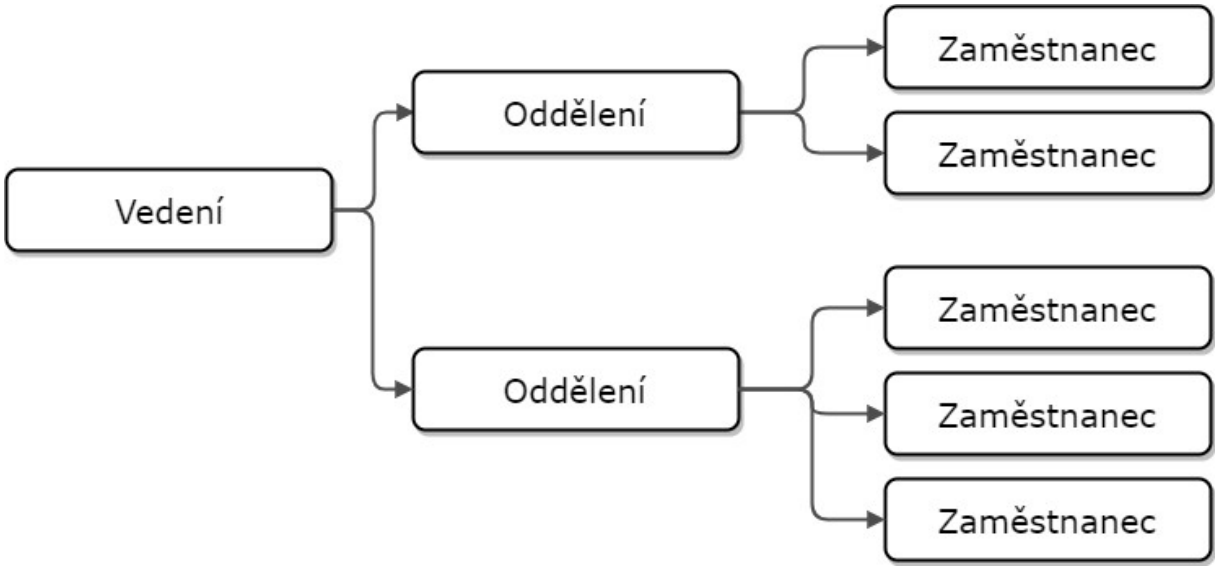
Pravidla, podle kterých se databáze tvoří, jsou nazývána jako „Systém řízení báze dat“ (DBMS v angličtině). Jinými slovy tvoří rozhraní mezi aplikačními programy a uloženými daty. Pointa těchto pravidel spočívá v tom pochopení pojmů, že databáze jako taková je kolekce dat, nýbrž „database management systém“ je způsob, jak získat k této databázi přístup a schopnost s ní manipulovat. DBMS by mělo splňovat tyto požadavky:

- Mělo by být schopno tvořit nové databáze a definovat jejich schémata. Nejčastěji se používá specializovaný jazyk DDL (data definition language)
- Provádění dotazů souvisejících s databází, jejich získávání a úpravu. Jazyk pro tento druh komunikace se nazývá DML (data manipulation language)
- Mít schopnost ukládat velké množství dat pro delší časový úsek. Dále zajistit efektivní přístup k takovým datům.
- Poskytovat trvanlivost produktu databáze, a poskytovat odolnost vůči chybám a výpadkům.
- Zajistit přístup k databázi z vícero uživatelských přístupů současně, aniž by byla narušena integrita dat, a vytvořena chyba či nežádoucí chování databáze.

S tímto definováním požadavků se začaly vyvíjet i jednotlivé architektury ukládání a zpracování dat. Od každého jiného druhu architektury se i odvíjí jiná pravidla DBMS. Těmito druhy architektury (v jiné literatuře pod názvem databázové

modely) mám na mysli model hierarchické databáze, síťové databáze, relační databáze, objektové databáze či objektově relační databáze.

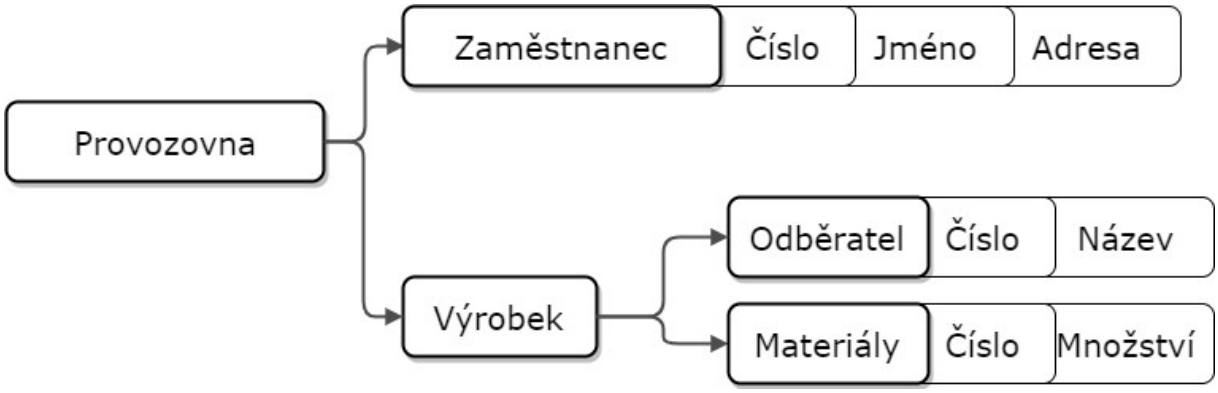
První datovým modelem, který měl komerční úspěch, byla **hierarchická** architektura. Již podle názvu řadí tento model data podle hierarchie, takzvané stromové struktuře. Průkopníkem tohoto modelu byla společnost IBM se svým systémem řízení dat IMS. Hierarchická koncepce má bohužel nedostatky při modelování databáze, která má odpovídat realitě. Postupně byla nahrazena koncepcí síťovou a relační. K popsání hierarchického modelu dat je potřeba si uvědomit, že ve stromové struktuře tohoto modelu se jedná vždy o vztah mezi uzly způsobem počátek/důsledek, jinou literaturou popsán jako rodič/potomek. K nalezení dat je potřeba se k nim vždy dostat přes proces předchůdců (počátků, rodičů). Tato skutečnost ztěžuje vkládání, opravu a rušení dat. Někdy může vzniknout nepřirozená organizace dat.



Obrázek 1 - Hierarchická architektura [vlastní výroba]

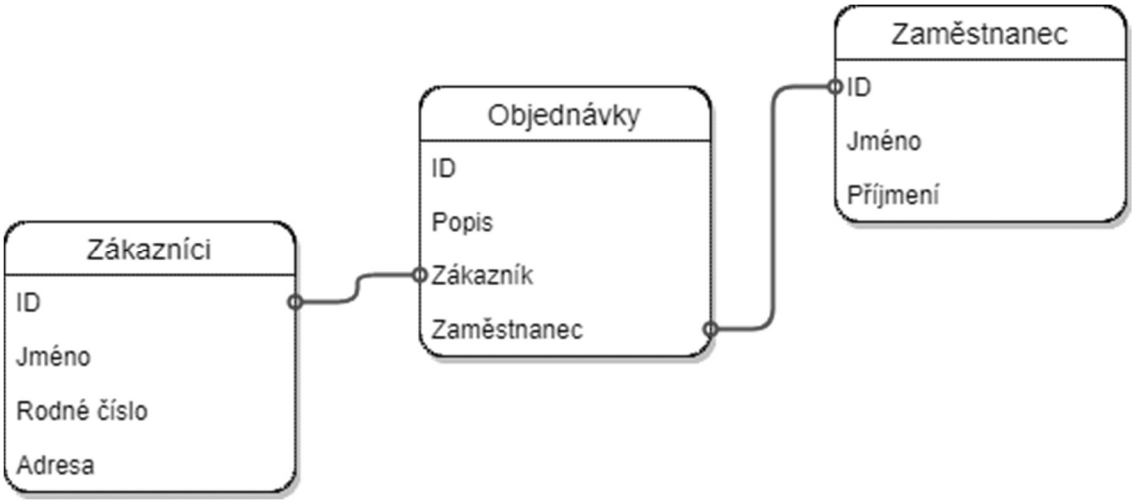
Nástupnickým modelem se stal **síťový** model. Je podobný hierarchickému, který byl do vývoje síťového dominantně využíván. Rozšiřuje hierarchický databázový model o další vztahy „více ku více“, což pro model znamená, že jeden důsledek může mít více předchůdců a naopak. K propojeným záznamům se přistupuje přímo bez dalšího vyhledávání, podle klíče. Vychází z hierarchického modelu a zdokonaluje ho, přímo na něj navazuje. Dominoval v komerčních databázových systémech 80. let 20. století. Jeho nejznámějším produktem byl IDMS od firmy Culliname Corporation.

Autorem této koncepce je C. Bachmann. Nevýhodou této síťové databáze je zejména nepružnost a obtížná změna struktury.



Obrázek 2 - Síťová architektura [vlastní výroba]

Revolučním modelem se stal model **relační**. Byl v roce 1970 popsán Dr. Edgarem F. „Ted" Coddem. Relační databázový model se po určité době stal nejpoužívanějším. Jeho struktura je organizovaná v tabulkách, které se skládají z řádků a sloupců. Tento vztah mezi skupinou prvků jedné nebo více tabulek (množin) matematicky nazýváme **relací.** V těchto tabulkách jsou prováděny veškeré databázové operace. Zajímavostí je, že i přes logickou praktičnost tohoto modelu nebyl v prvních letech po publikování přijat. IBM, i vzhledem k tomu, že hardware v té době nebyl tolik výkonný, sázelo na starší databázové struktury. Až tím, že E. F. Codd nabízel tento model přímo zákazníkům a konkurenčním společnostem (Oracle, Sybase), se tento model začal plně využívat, což vedlo k posunu ve vývoji počítačové technologie.



Obrázek 3 - Relační architektura [vlastní výroba]

Relační databáze musí splňovat dvě základní vlastnosti:

- Databáze je chápána jako množina relací, nic víc, nic míň.
- V relačním systému řízení báze dat jsou k dispozici minimálně operace: selekce, projekce a spojení. Tyto operace si nevyžadují jasně předdefinované přístupové cesty pro realizaci těchto operací.

Dále Codd definoval dalších 12 pravidel pro relační SŘBD:

1. Pravidlo informační
    - Všechny informace v relační databázi jsou zapsány na logické úrovni jediným způsobem, a to hodnotami v tabulkách.
2. Pravidlo jistoty
    - Všechna data v relační databázi jsou jednoznačně přístupná díky kombinaci jména tabulky s hodnotami primárního klíče a jméno sloupce.
3. Systematické zpracování nulových hodnot
    - Nulové hodnoty jsou plně podporovány pro reprezentaci informace, která není definovaná, a to nezávisle na datovém typu.
4. Dynamický online katalog vycházející z relačního modelu
    - Popis databáze je vyjádřen na logické úrovni stejným způsobem jako zákaznická data, takže autorizovaný uživatel může aplikovat stejný relační jazyk ke svému dotazu jako uživatel při práci s daty.
5. Obsáhlý datový podjazyk
    - Relační systém může podporovat několik jazyků a různých módů použitých při provozu terminálu. Nicméně musí být nejméně jeden příkazový jazyk s dobře definovanou syntaxí, který obsáhle podporuje definici dat, definici pohledů, manipulaci s daty jak interaktivně, tak programem, integritní omezení, autorizovaný přístup k databázi, transakční příkazy a podobně.
6. Pravidlo vytvoření pohledů
    - Všechny možné pohledy jsou systémem teoreticky vytvořitelné.
7. Schopnost vkládání, vytvoření a mazání

- Schopnost zachování relačních pravidel u základních i odvozených relací je zachována nejen při pohledu na data, ale i při operacích průniku, přidání a mazání dat.

8. Fyzická datová nezávislost
    - Aplikační programy jsou nezávislé na fyzické datové struktuře.
9. Logická datová nezávislost
    - Aplikační programy jsou nezávislé na změně v logické struktuře databázového souboru.
10. Integritní nezávislost
    - Integritní omezení se musí dát definovat prostředky relační databáze nebo jejím jazykem a musí být schopna uložení v katalogu, a nikoliv v aplikačním programu.
11. Nezávislost distribuce
    - Relační SŘBD musí být schopny implementace na jiných počítačových architekturách.
12. Pravidlo přístupu do databáze
    - Pokud je relační systém napsán v jazyce nízké úrovně, nemůže tento jazyk být použit k vytvoření integritních omezení a je nutné použít relační jazyk vyšší úrovně.

Relační databázový model má dodnes velký úspěch a jedná se o nejpoužívanější databázový model. I vzhledem k tomu, že jsou relační databáze tak rozšířené, bylo potřeba vytvořit sadu příkazů pro jejich ovládání. Prvním takzvaným jazykem byl SEQUEL (Structured English QUEry Language), jehož cílem bylo přiblížit příkazy pro tento typ databáze přirozenému anglickému jazyku. Tato počáteční verze jazyka byla postupem času upravována, a to nejen tvůrcem, tedy společností IBM, ale i dalšími subjekty jako „Relational software, Inc.“ (dnešní Oracle). V průběhu let se vytvořily verze SQL jazyka:

- SQL-86 (známý též jako SQL-1) od Amerického institutu ANSI
- SQL-89 jakožto menší revize SQL-1
- SQL-92 (známý též jako SQL-2) jakožto velká revize od ANSI a ISO. Tato verze měla největší komerční úspěch

- SQL:1999 (známý též jako SQL-3), což je významně upravená a rozšířená verze jazyka, která přidala funkce spojené s objektově relačním modelem databáze.
- SQL:2003 je jazykovým standardem, který umožňuje aplikovat XML-funkce.

Práce s relační databází vyžaduje systematičnost a znalost problematiky. S rozšiřujícími technologiemi a postupy v programování začaly vznikat **objektově** orientované databáze, které si kladou za cíl ulehčit a zrychlit práci s daty. Počáteční nadšení s prací s objektovým modelem databáze postupem času upadalo, neboť objektový přístup je oproti relačnímu rozdílný, a navíc k tomu velmi náročný nejen na představivost. Objektové databáze nemají žádný oficiální standardizovaný jazyk na rozdíl od relačního modelu. Objektové databáze kombinují prvky objektově orientovaného programování s databázovými schopnostmi. Rozšiřují funkčnost objektových programovacích jazyků (C++, Java) a poskytují plnou schopnost programování databáze. Datový model aplikace a datový model databáze se ve výsledku hodně shodují a výsledný kód se dá mnohem efektivněji udržovat.

Postupným vývojem a vyššími nároky se objevil nový model, takzvaný „**rozšířený relační**“ nebo „**objektově-relační**“ model. Snahou tohoto modelu je sjednotit rysy relačních a objektových databází. Způsob tvorby ORDBMS (Object-related database model systém) je zakomponován ve SQL standardu SQL-3. Funguje to takovým způsobem, že jsou relační tabulky rozšířeny o „objektovost“. Všechny trvalé informace jsou obsaženy v položkách tabulek, ale některé položky mohou mít bohatší datovou strukturu o takzvaný abstraktní datový typ. ADT vznikne zkombinování základních datových typů. Základní nevýhodou tohoto modelu je jeho omezená rozšiřitelnost pro nové datové typy, nezralost a neprozkoušenost.

2.3 Multidimenzionální databáze

V rámci databází je často potřeba vyhledávat data z různých perspektiv. Existuje tedy efektivní způsob, jakým systém řízení báze dat toto vyhledávání podporuje? Multidimenzionální databáze a multimediální databáze na tuto otázku nalézají odpověď.

Pod pojmem „multidimenzionální data“ se skrývá význam spojený s tématem řízení dat. Tento pojem se dá pochopit jako data souhrnných ukazatelů, které jsou

vytvořeny různým shlukem relačních dat, jež jsou určeny pro on-line analytické zpracování (OLAP). „Online analytic processing“ je software pro podporu rozhodování. Informace v něm obsažené jsou shlukovány do multidimenzionálních pohledů a hierarchií. Snaha výpočetního předzpracování dat vede k urychlení příkazů a ke konsolidaci informací z různých databází do takzvaného datového skladu. Jako příklad se můžou uvést stylizované příkazy, jakými jsou: seskupení, sloučení, statistické funkce či analýzy časové řady.

OLAP je tedy technologií, která umožňuje uspořádat velké objemy dat tak, aby byla data přístupná a pochopitelná uživatelům, kteří se zabývají jejich analýzou. Jeho alternativou je OLTP (Online transaction processing), který klade důraz na snadné a bezpečné uložení změn v datech, a to v mnohauživatelském prostředí.

Pokud se ale budeme držet technologie OLAP, tak ta upravuje data do dimenzí (hierarchicky uspořádaných hodnot), a posléze je strukturuje do multidimenzionální datové kostky. Tato **datová kostka** je obecně přijímanou základní logickou strukturou, která definuje **multidimenzionální databáze** tak, jak definuje relace relační databázi. Stejně tak jsou multidimenzionální databáze vymezeny 12 pravidly, které definoval E. F. Codd a jsou uvedeny výše.

2.4 Proces návrhu relační databáze

Správně navržená databáze přináší uživateli přístup k aktuálním a přesným informacím. Vzhledem k tomu, že je správný design nezbytný ke správnému a úplnému využití práce s databází, je smysluplné se zabývat principy dobrého návrhu databáze. Nakonec každý uživatel uvítá práci se snadno pochopitelnou, uživatelsky příjemnou databází. Proces návrhu databáze se skládá ze tří po sobě jdoucích návrhů:

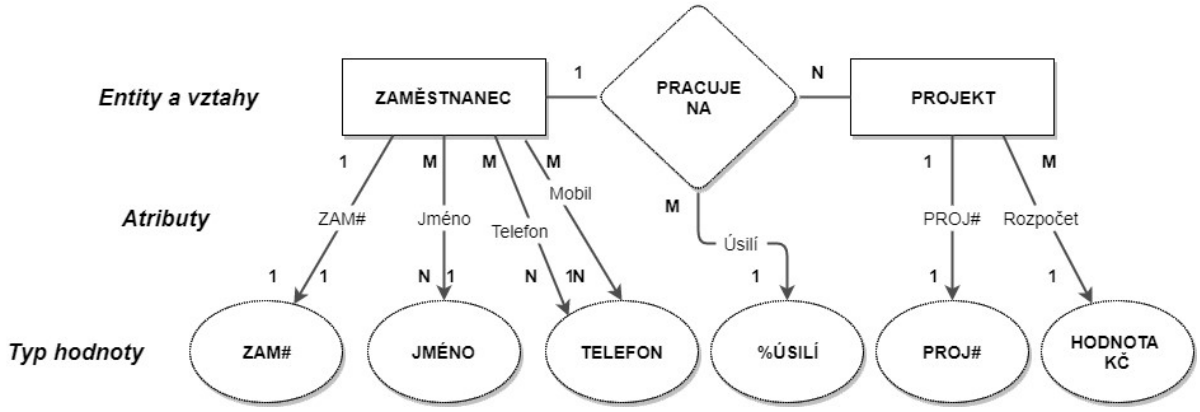
1. Konceptuální návrh (model reality)
2. Logický návrh
3. Fyzický návrh

Takto nastavené návrhy vedou návrháře k systematické realizaci databáze tím, že se těmito postupnými kroky vytříbí myšlenka projektu a analyzují se požadavky na úspěšnou realizaci.

2.4.1 Konceptuální návrh

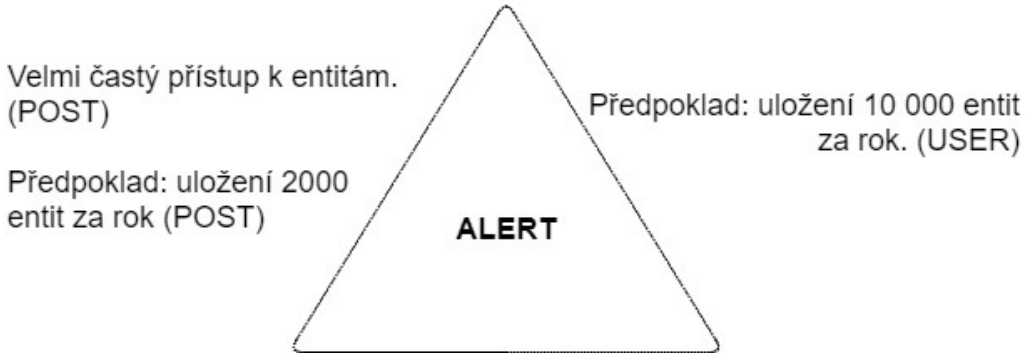
Způsob, kterým se databáze v první fázi navrhují, se nazývá konceptuální modelování. Jedná se o fázi datové, případně objektové analýzy, která využívá modely založené na objektech, které symbolizují jednotlivé části reality, které jsou pro danou databázi důležité. Všechny vztahy těchto objektů je třeba věrně popsat. V této fázi se vytvoří takzvaný **model reality,** který popisuje obsahovou stránku. Ke správnému popsání této konceptuální úrovně návrhu je potřeba popisovat určité procesy i mírně abstraktně, ale hlavně relevantně. Zcela jsou ignorována technologická či implementační specifika návrhu. Cílem modelu reality je zjednodušený pohled na obvykle příliš široké a složité reality. Z tohoto modelu vycházejí další dva; technologický a implementační. Vzhledem k tomu je nutné v konceptuálním modelu popsat vše, co je popsané pro každého člena realizačního týmu databáze. Je nevhodné v dalších fází realizace zanedbat či si domýšlet určité části návrhu. V konceptuálním modelu reality se tedy řeší, CO chceme mít v systému, ale NE co je řešením pro zprostředkování tohoto systému. Velmi často se v tvorbě modelu reality používají ER diagramy.

ER modely či diagramy byly publikovány Peterem Chenem v březnu roku 1976. Lze je použít jako základ pro sjednocení odlišných pohledů na data, pohledů, kterými jsou na mysli síťový model, relační model či model množiny entit. Byl inspirován skutečným vnímáním reality. Pro grafické znázornění ER modelu se používá ER diagram. Technikou ER diagramů je zobrazení typů entit (základních objektů), typů jejich vztahů a atributů. Entita je základním skutečným objektem, který je charakteristický, odlišný od jiných objektů. Množinou entit je namysli kolekce podobných entit, kterými můžeme mít na mysli ku příkladu množinu zaměstnanců. Všechny entity jsou popsány pomocí hodnot. Všechny entity v množině entit musí mít stejnou množinu hodnot (všichni zaměstnanci musí mít definovatelné hodnoty-atributy, kterými jsou ku příkladu: jméno, id číslo, telefon, adresa aj.). Hodnoty mohou být chápány jako funkce mapující entity a zařazuje jim definiční obor.



Obrázek 4 - ER diagram [podle Chena, 2002]

Alternativním modelem pro řešení konceptuálního návrhu je ER-A model, který je inovovanou verzí předchozího Chenova ER modelu. Rozšiřuje tento model o množinu upozornění, což je de facto informace o entitách, které by jinak byly zkreslené nebo nedostupné. Tato množina se nazývá Entity Set of Alerts (ES-A) a graficky se vyjadřuje pomocí trojúhelníku, názvy atributů se obecně vyjadřují v závorkách, které jsou připsány ke každé hodnotě.



Obrázek 5 - Množina upozornění ER-A [podle Tyrychtr, a další, 2010]

Tento inovativní způsob konceptuálního modelu dat si klade za cíl zlepšit kvalitu databázových projektů, a to pomocí toho, že se snaží eliminovat komunikační šum přenosu informací mezi jednotlivými členy vývojového týmu. Takovýto model s pomocí ER-A diagramu se nazývá Entity-Relationship Alert (ER-A) model.

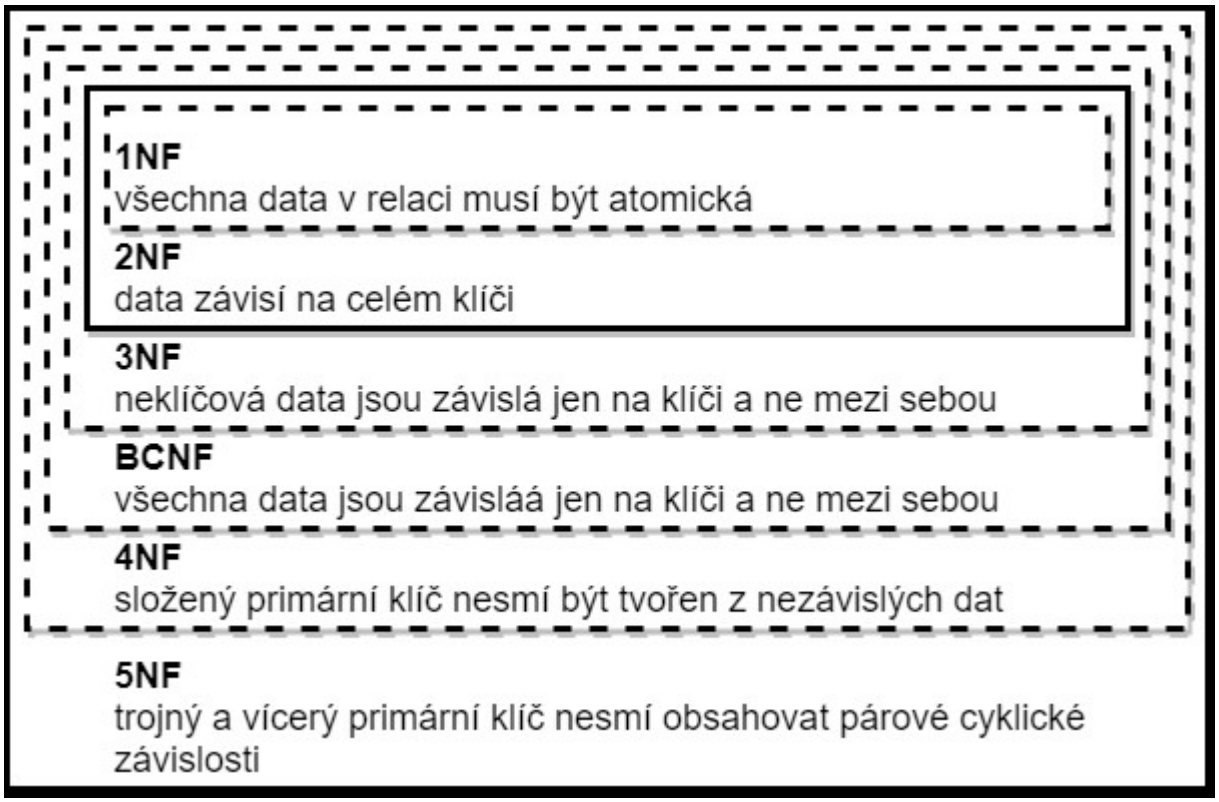
2.4.2 Logický návrh

Na konceptuální úroveň navazuje logický návrh (někdy také zvaná jako technologická úroveň) představující střední míru abstrakce. Jedná se o návrh, který vychází z konceptuální úrovně a rozšiřuje ho o technologické řešení problému. Zde se určuje, jak budou data strukturovaná (použije-li se objektová databáze, relační

databáze apod. pro řešení problému) a jak bude tato navržená technologie realizovaná. Berou se v úvahu dostupné technologické prostředky a možnosti dané architektury, ale nezachází se do detailů a specifik řešení. Ta jsou řešena až v poslední fázi. Logická úroveň popisuje **JAK** bude obsah vypadat, jak bude vymezený a jak bude realizován. Je teoreticky možné, že pro jeden konceptuální návrh bude řešením více různých logických modelů, neboť jeden návrh je možné realizovat více způsoby.

Návrh logického schématu často zahrnuje normalizační kroky, které spočívá v přenosu logického schématu na normalizované schéma za použití normálních forem. Normální forma definuje stav závislosti množin dat, nebo sémantické omezení zadané jako součást databázového schématu. Tyto podmínky se používají ke kontrole, zda návrh databáze má požadované vlastnosti. Tyto normální formy jsou k nalezení v řadě publikací, podobně jako normalizační algoritmy, které převádí špatně navrženou databázi do správně navržené normální formy databáze. Normální formy mají tento popis:

Tabulka 1 - Popis normální formy [vlastní výroba]



2.4.3 Fyzický návrh

Fyzická fáze návrhu je odlišná koncepcí, neboť se zaměřuje na efektivitu výsledné databáze z hlediska funkčnosti. Cílem je mapovat logické schéma databáze

tak, aby bylo zohledněno příslušné úložiště v databázovém systému, včetně fyzických parametrů k efektivnímu výkonu databáze pro práci s transakcemi.

Fyzický návrh je třetím krokem v návrhu databáze, kde se předchozí logický návrh převede na fyzický, který je použitelný v rámci zvoleného databázového systému. Každý databázový systém má svá specifika, z čehož plyne, že může existovat vícero postupů v rámci fyzického návrhu databáze, které reflektují specifika jednotlivých databázových systémů. Je vhodné mít při řešení fyzického návrhu znalost jiných databázových systémů a při fyzickém návrhu je potřebné uvažovat i o tom, zda by změna databázového systému nepřinesla některé výhody.

Pokud bychom srovnávali logický návrh s fyzickým, tak logický návrh je založen na specifickém modelu dat (relační, objektově-relační či jiný model), ale nebere v potaz podrobnosti spojené s požitím konkrétního databázového systému. Výstupem logického modelu je popsaná množina relací tabulek. Fyzický návrh ale pracuje s tímto logickým modelem v rámci cílového databázového systému tak, aby pochopil jeho funkce a možnosti. Fyzický model jinými slovy je do značné míry odrazem individuálních vlastností cílového databázového systému. Je také možné, že na základě fyzického návrhu se pozmění i předchozí logický návrh, a to třeba z důvodu zjištění, že jiné rozložení tabulek daného databázového systému bude výkonnější.

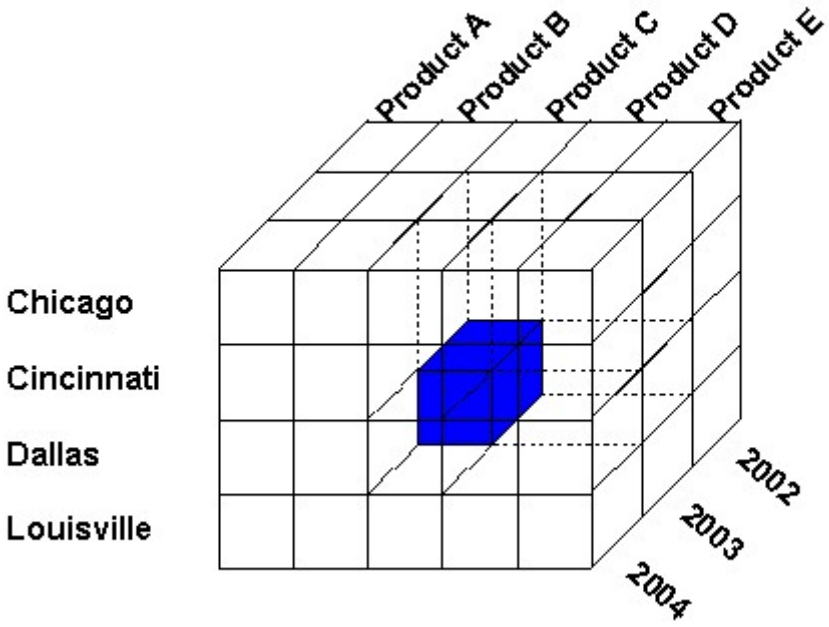
Fáze fyzického modelování se skládá z následujících činností.

- Přenesení logického návrhu databáze do návrhu vyhovujícímu cílovému databázovému systému
- Volba indexů a organizace souborů
- Návrh zabezpečení a jeho mechanizmů
- Spuštění systému do provozuschopného stavu

2.5 Proces návrhu databáze

Multidimenzionální modelování je proces strukturování dat pomocí modelovacích konstrukcí, který vyústí ve vícerozměrný model dat. Multidimenzionální modely kategorizují data jako fakta asociovaná s numerickou hodnotou, nebo jako dimenze, která bývají spíše textového charakteru. Fakty jsou na mysli objekty, které představují předmět požadované analýzy, který má být analyzován pro lepší pochopení jeho chování. Multidimenzionální modely v současné době nejčastěji

vycházejí z relačního modelu dat, popřípadě jsou založeny na multidimenzionální datové kostce, která představuje věcně orientovaná data.



Obrázek 6 - Multidimenzionální kostka, [1]

Multidimenzionální model dat vycházející z relačního modelu odlišuje dva základní typy relací, které se nazývají tabulky dimenzí a tabulky faktů. Oba typy tabulek jsou v zásadě databázové relace s určitými specifiky, které zohledňují cíl, pro který jsou určeny. Tyto relace mohou být strukturovány do hvězdicových struktur, formy sněhové vločky či souhvězdí.

Multidimenzionální model dat by měl být sestaven v souladu s návrhem BI systému společnosti (Bussiness inteligence systém), což je návrh dovedností, znalostí, technologií, kvality, rizik a postupů v podnikání. Jinými slovy každý model dat musí být vhodně postaven pro potřeby daného podniku. Jedním způsobem, jak tento BI systém nastavit, je tento postup:

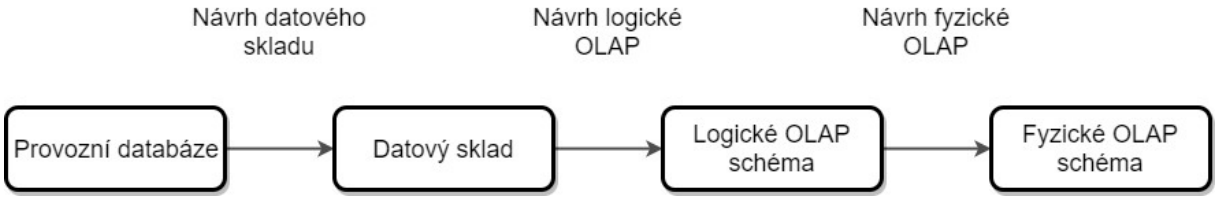
1. Analýza potřeb uživatelů pro práci s informacemi
2. Analýza datové základny
3. Návrh řešení a architektury

Multidimenzionální modelování vyžaduje specializované návrhové metody. Stále totiž neexistuje jednotná metodika návrhu i přes to, že je mnoho napsáno o

návrhu datových skladů. Jedním z přístupů může být použit ten první Kimballův, který doporučuje čtyři kroky v procesu multidimenzionálního modelování:

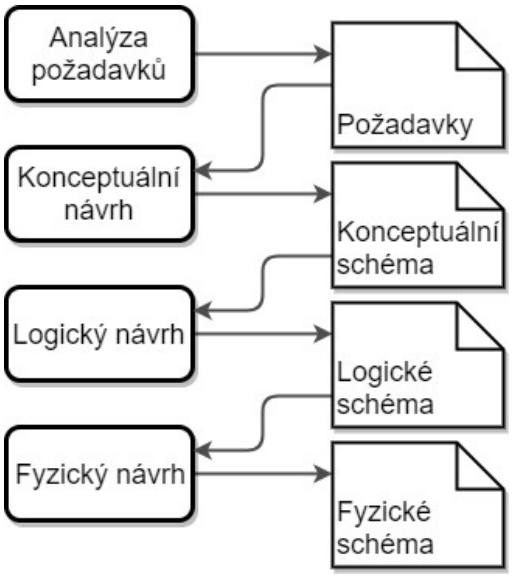
1. Výběr podnikových procesů
2. Výběr části z podnikového procesu
3. Výběr rozměrů (dimenzí)
4. Výběr měr

Jiný přístup počítá s procesem návrhu OLAP a datového skladu podle Niemiho:



Obrázek 7 - Proces návrhu OLAP databáze [2]

Je potřeba napsat, že pro multidimenzionální modelování je potřeba použít specializované návrhové metody. Další metoda návrhu datových skladů níže prezentovaná vychází z diskuze IT odborníků, a byla prezentovaná v Rizziho článku z roku 2006.



Obrázek 8 - Základní fáze návrhu datového skladu [3] [4]

Všechny tyto metody návrhů dílčích kroků jsou systematizovány a prezentovány v literatuře spojené s tématem databázových systémů. Pro pokračování v práci bude využit model návrhu datového skladu.

3 Shrnutí předchozí kapitoly

Celé poměrně stručné shrnutí historie a tvorby databáze, které je obsahem předchozí kapitoly, vede k seznámení se s problematikou. Teorie v předchozí kapitole popsaná představuje postupy, které se vývojem databázových systému vyvinuly. Jedná se o zavedené postupy, které vedou k usnadnění a systematičnosti práce na tvorbě databází.

Pro další práci je potřeba definovat primární entitu, na které se všechny další vlastnosti databáze takzvaně nabalí. Vzhledem, že tato práce se zabývá evidencí majetku, právě tento **majetek** musí být primární entitou. Této entitě se musí přiřadit atributy, které tento majetek definují, jakými mohou býti například evidenční číslo, hodnota, stáří nebo jiné vlastnosti.

K této primární entitě, která je podstatou práce, se přidají další entity, které reprezentují prostředí stavebního podniku. Těmito entitami jsou namysli ku příkladu lokalita, číslo projektu (zakázky), zodpovědná osoba či číslo úvěru.

Dalším krokem v této práci je návrh databáze. Jak je zmíněno výše v předchozí kapitole, správný a systematický postup při návrhu se skládá ze tří na sebe navazujících kroků. Prvním krokem je konceptuální návrh neboli model reality, který hrubě nastíní, jak by mohla evidence vypadat. Dalším krokem je logický návrh, ve kterém se pracuje se vztahy jednotlivých entit a celý návrh databáze získává reálné obrysy. Třetím krokem je na mysli fyzický návrh, který k logickému návrhu přidá i technické požadavky na databázi jako takovou.

4 Problematika evidence majetku

4.1 Mechanizace stavební společnosti

Mnoho stavebních dodavatelů investuje podstatnou část peněz do vybavení stavebního podniku, ať už do strojů, zařízení dílen a jiných hmotných aktiv. Kontrola faktického a vnitropodnikového účetnictví mechanizace je obzvlášť důležitá vzhledem k tomu, jaké objemy peněz se v těchto aktivech skrývá. Pro management podniku je také důležité vědět, jak ocenit vytíženost a využití strojů na jednotlivých projektech, jež stavební podnik dodává.

Je tedy zřejmé, že evidenci veškeré významnější mechanizace podniku je nutné zavést a udržovat. Měla by fungovat kontrola a zaznamenávání pohybů vybavení mezi jednotlivými projekty, a to z průběžného hlášení těchto pohybů vedoucími projektů (stavbyvedoucími a vedoucími skladů). Periodické zprávy by měli zaznamenávat lokalitu vybavení pro usnadnění přehledu pro projektové manažery. Mechanizace v terénu (na stavbách) je nutné pravidelně podrobit inventarizaci, obzvlášť při dokončování projektu a během nástupu nového účetního období. K vybavení by měly být přiřazeny náklady s ním spojené, které odpovídají účetním záznamům tak, aby mohl dodavatel adekvátně ocenit použití tohoto vybavení na jednotlivém projektu. Malé vybavení, kterým máme na mysli ruční nářadí, bývá naceněno přímo k zakázce. Tyto postupy by měli vést ke kontrole na pracovištích, která zamezí ztrátám a rozkrádání.

Z pohledu vyššího managementu společnosti je výhodné, aby veškeré podnikové vybavení bylo evidováno i v zájmu účetního odpisování. Mnohé stavební stroje jsou obzvlášť u menších podniků využívány déle, než uvádí jejich odpisová životnost. Opětovné nacenění těchto strojů znalcem může významně navýšil hodnotu majetku podniku. U velkých podniků se této skutečnosti u strojů tolik nevyužívá, neboť pro velký podnik může být výhodnější mít majetek (automobily, stroje, ruční nářadí) smluvně pronajaté většími pronajímateli.

Na jednotlivé projekty je vhodné naplánovat využití mechanizace. Plán mechanizace je důležitý pomocník k efektivnímu využití vybavení v čase a místě tak, jak bude potřeba. Co nejpřesnější řešení plánu mechanizace vede k efektivnímu využití vybavení podle potřeby a podle dostupnosti. Některé práce ve stavebnictví jsou úzce spjaty s ročním obdobím, jsou takzvaně sezónní. Vhodné plánování

mechanizace, která je úzce spjatá se sezónními pracemi, vede k větší efektivitě, a to jak finanční (nebude potřeba zapůjčení jednoho typu stroje kvůli využívaní toho typu na jiném projektu), nebo časové (zamezí se prodlení z důvodu absence typu stroje). Navíc vhodné plánování vede k lepšímu řešení případných změn projektu (vícepráce, méněpráce), jejich ocenění, načasování a navazování na další činnosti. Správná evidence mechanizace vede k tomu, že náklady na mechanizaci jsou vyúčtovány na stavbě, kde daná mechanizace skutečně vykonává práci.

Dostatečná evidence mechanizace také může vést ke zjištění, že některé typy mechanizace jsou nadužívané a některé naopak dlouhodobě nevyužívané. Tato informace může vést k poznatku, že bude výhodnější investovat do další mechanizace. Naopak každý majetek podniku, který stojí, je nevyužíván, ztrácí postupně na hodnotě a negeneruje obrat, je tedy spíše na obtíž. Jediné jeho využití je pro případně krytí při požadovaném úvěru. Úvěrové společnosti si ale v tomto případě velmi vybírají, čím chtějí mít své poskytnuté úvěry kryty.

4.2 Náklady mechanizace

Jak se náklady na materiál a lidské zdroje mění v čase a v závislosti na změně projektů, tak podobně se mění náklady na materiál. Tyto náklady mají několik proměnných. Jasnou první proměnou je čas, jak dlouho mechanizace plnila svojí funkci, vynásobená hodinovou sazbou. Tyto náklady mechanizace se standardně v ČR spočítají z norem pracovních prostředků. Musí se během výpočtu výkazu výměr provedených prací rozlišit, co je vícepráce, nebo co je nadbytečný výkon. Úhrada víceprací se požaduje po zadavateli, který změnil v průběhu realizace projekt, zatímco nadbytečný výkon či sanace jdou čistě za dodavatelem. Některé další náklady na vybavení mohou vyústit z nákladů na pronájem mechanizace, která byla způsobena změnami v projektu, neboť dodavateli může oproti původnímu plánu chybět mechanizace potřebná k realizaci. Dále mohou nastat náklady s opětovným zřízením a odstraněním pracovní mechanizace, a v neposlední řadě se musí brát v potaz i náklady stojícího, nepracujícího stroje.

Pro dodavatele jsou stěžejní dokumenty, jakými jsou rozpis a plán prací, včetně soupisu provedených prací, které jsou důležitými zdroji pro zjištění hodnoty nákladů jak minulých, tak současných. Dále by měl mít dodavatel soupisy plánovaných a provedených prací každého stroje na stavbě. Důležitá je i informace o tom, zdali plán

mechanizace odpovídá realitě. Součástí soupisů prací by měl být každé vybavení na stavbě použité, tedy třeba i starý pickup, který pomalu dosluhuje. Je třeba počítat s tím, že jednotlivé projekty vždy mají různě vypočítané náklady na mechanizaci. Byť je to samozřejmě náročnější, je jednoznačně lepší počítat náklady na mechanizaci přímo ze soupisu provedených prací, než je zjišťovat třeba procentuálně tak, jak se počítají nepřímé náklady. Soupis provedených prací stroje je stejně podstatný pro výpočet nákladů stroje jako je stavební deník podstatný pro realizaci celé stavby. Soupisy provedených prací stroje se v praxi nazývají různě, u větší mechanizace se zapisují například do „knihy výkonů“, „deníku výkonů“ nebo do „provozní knihy“. Pokud je zájem evidovat i menší mechanizaci, jakou je třeba ruční mechanizované nářadí, je více než vhodné sledovat pohyby mezi stavbami této mechanizace z výdejových listů. Náklady se poté spočítají na stavbu z doby, mezi vydáním mechanizace na stavbu a poté její návrat či přesun na jinou.

Zpoždění projektu mohou vést k nepokrytým nákladům na mechanizaci, neboť výkonové normativy počítají s určitou aktivitou a výkonem. Tyto nepokryté náklady z prodlení je možné pokrýt, jestli bylo zpoždění způsobeno zadavatelem či třetí stranou.

4.3 Druhy majetku stavebního podniku

Pro pochopení, co vlastně se bude evidovat, je potřeba strukturalizovat majetek. Nejlepším způsobem, jak toho dosáhnout, nabízí základní účetní rozdělení aktiv (tedy majetku) v rozvaze. Základně se majetek tedy dělí podle doby využití na:

- Dlouhodobá aktiva, jež jsou používána déle než rok
- Krátkodobá aktiva, která mění svou podobu v průběhu účetního období, tedy jednoho roku, též nazývaná oběžná aktiva

Tento způsob dělení je vhodný pro pochopení funkcí majetku. Dlouhodobý majetek má pořizovací hodnotu, která se během doby mění. Dělí se na nehmotný majetek, hmotný a finančně investiční majetek. Krátkodobý majetek se dělí na zásoby, pohledávky, krátkodobé investice a patří pod něj i peníze.

Z pohledu evidence majetku ve stavební společnosti je potřeba zejména evidovat ty položky, které mají jednak podstatnou finanční hodnotu, a dále ten majetek, který se pohybuje po jednotlivých stavbách či je používaný na jednotlivých projektech.

Je vhodné mít díky evidenci přehled o tom, jakou flotilu mechanizace má podnik k dispozici z vlastních zdrojů, aby zároveň díky tomu mohl i poupravovat technologie realizace projektu.

Mezi taková aktiva zejména patří ta hmotná. Účetně se jedná o majetek, který měl pořizovací hodnotu nad 40 000 Kč a jeho životnost je delší než 1 rok. Jedná se mimo jiné o pozemky, stavby, samostatně movité věci a soubory movitých věcí (stroje, zařízení). Pozemky a stavby jako takové neplní ve stavebním podniku tolik funkci finančně-oběhovou, pokud tedy není záměrem pozemek stavebně zhodnotit a prodat. Touto developerskou činností se ale evidence majetku nebude zabývat. Podstatou evidence jsou stroje a zařízení, neboť jejich efektivnost v realizacích potřebuje dodavatel podchytit.

Velké stroje, podzemní mechanizace a dopravní prostředky jsou položky, které mají největší význam ve stavebním podniku. Možnost využití těchto strojů efektivně je jedním ze zásadních prostředků, jak akcelerovat ziskovost podniku. Evidence strojů, které má podnik k dispozici, a co nejpřesnější plánování zařizuje podklad pro přípravu zakázky a její ziskovost. Díky evidenci a plánování se může zamezit zbytečným výdajům na pronájem mechanizace od třetí strany. Dále z evidence je vhodné vidět i dobu, kdy a kde daný stroj byl používán či umístěn.

Je vhodné evidovat in nehmotná aktiva. Není v silách člověka si pamatovat všechna know-how, pořízené softwary a licence, případně patenty, ochranné známky a jiná povolení, které má podnik ve vlastnictví. Obecně se jedná o aktiva pořízená za 60 000 Kč a použitelná déle než rok. Ve středních a malých podnicích je stále časté použití nelegálních kopií softwaru i přes to, že již má podnik daný software zakoupen, jen o tom uživatelé (zaměstnanci) nejsou informováni.

Co se týče oběžných aktiv, zejména se jedná o evidenci malých mechanizací, v případě stavebnictví o ruční nářadí. To se obvykle nacení přímo k dané zakázce. Některé ruční nářadí, zejména elektrické, se reálně použije na více projektech. U tohoto druhu náčiní je tedy evidence vhodná, zejména kvůli zamezení krádeží. Je vhodná i evidence spotřebního nářadí, jakými jsou třeba lopaty aj. Zde ovšem stačí pouze souhrnně spočítat počet vydaných kusů na danou stavbu či danému pracovníkovi. Jedná se v celém obratu stavební firmy o více méně marginální položky, leč i sledování trendů v této položce může leccos naznačit o efektivitě na daném

pracovišti či projektu. I psychologický vliv na pracovníky, že práce je organizovaná, nástroje jsou dostupné a jsou evidovány a sledovány toky vybavení, je může vést k větší spokojenosti v zaměstnání, nebo naopak tato skutečnost zamezí případným ztrátám.

Evidence materiálu je také potřebná vzhledem k možnostem ušetření výroby. Základním a nejvýhodnějším způsobem, jak nakládat s materiálem, je způsob „just-in-time“. Jedná se o metodu nákupu materiálu podle aktuální potřeby bez meziuskladnění, s odvozem přímo na stavbu. K tomuto řešení potřeb obratového materiálu je třeba mít dobře naplánovanou potřebu materiálu a také nejlépe mít předjednanou smlouvu o odběru u některého dodavatele.

Zároveň je vhodné evidovat materiál, který byl již použit na jiné stavbě. Tímto materiálem je na mysli ten materiál, který byl vyzískán při realizaci jiné stavby. Příkladem takovéto stavby mohou býti štětovnicové stěny, které jsou realizovány při výkopech ve vlhké půdě či přímo v tocích řek. Rozpočtově jsou štětovnice naceněny jako celek, reálně se ovšem po realizaci stavby štětovnice vytahují (čemuž se neodborně říká výzisk, který v případě rabování štětovnic dosahuje odhadem 80%) a použijí znovu na jiné stavbě. Tam jsou štětovnice naceněny znovu. S logickým přístupem ke koupi materiálu „just-in-time“ se může ušetřit i díky evidování materiálu dříve vyzískaného. Je ovšem mít potřeba mít na paměti, že dlouhodobé skladování na deponii minimálně zabírá místo případnému potřebnějšímu majetku, a je tedy potřeba takto s tím pracovat.

Shrnutím kapitoly zjistíme, že evidence majetku by měla sledovat pohyby:

- Velkých strojů, včetně plánování a zpětné sledování jejich využití
- Malé mechanizace, ruční nářadí
    - Elektrické ruční nářadí, a to i s lokalitou, kde se nachází
    - Spotřební náčiní kumulativně (lopaty, rukavice), tedy počty vydané na stavbu či četě
- Licencí, softwaru pro sledování jejich počtu a rozsahu.

4.4 Požadavky na evidenci z pohledu uživatelů

Evidence by měla být jak fakticky správně vedená, tak i přehledná. Evidence by měla plnit různé funkce pro různé pracovníky podniku. V podstatě se jedná o tři různé

úrovně možného přístupu k evidenci majetku. Tyto úrovně jsou rozděleny podle pracovníků, kteří s touto evidencí pracují. Těmito pracovníky jsou majitel nebo manažer, stavbyvedoucí a skladník.

Majitel, manažer či účetní jsou pracovníci, kteří budou požadovat po evidenci zjištění o stáří, hodnotě majetku, lokalitě a využívanosti. Jejich přístup k těmto informacím by měl být úplný, na základě těchto informací se rozhodují o nákupu, prodeji či o tom, zdali je podnik schopen svojí mechanizací realizovat nabízené zakázky. Zároveň je vhodná archivace dat pro vyčíslení nákladů na zakázku či pro účetní důvody. Z celkové evidence může pracovník na manažerské pozici řešit případné nedostatky.

Vedoucí projektů, stavbyvedoucí či mistři by měli mít možnost se orientovat v evidenci tak, aby mohli majetek alokovat ke svým potřebám. Je pro něj významnou informací, když ví, že je pro něj potřebná mechanizace v podniku, a měl by vyslat požadavek pro jeho použití. Tento postup šetří náklady na realizaci. Tento vedoucí pracovník by měl velmi úzce spolupracovat se skladníkem a měl by vést přesnou evidenci pohybu na svých stavbách.

Skladník jakožto pracovník, který spravuje zásoby či nevyužívanou mechanizaci, by měl vést jejich evidenci na skladu, či deponii. Veškeré nákupy nové mechanizace či naopak jejich odepisování, chcete-li vyhazování, by měly jít přes něj, čímž tyto nákupy, prodeje či odepisování zapisuje do evidence.

K efektivnímu vedení evidence je vhodné použití kódů pro každou položku evidence. K urychlení evidování se občas používá skener čárových kódů či čipování, zejména v oblasti velkoobchodu a maloobchodu. Tímto postupem bude mít každá jednotlivá položka evidence jedinečný kód, který se při vyhledávání v databázi použije a urychlí veškeré potřeby v souvislosti s evidencí.

4.4.1 Kategorizace majetku

Dále je důležité zvolit rozřazení majetku podle kategorií. Kategorizace je podstatná pro logické zařazení majetku do evidence. Pro zvolení kategorií ve stavebním podniku je potřeba uvažovat podle zaměření společnosti, samotnému majetku ve společnosti a podle subjektivní logické úvahy uživatele. Tím je namysli použití kategorií, kterému přijdou každému uživateli evidence logické. Pokud bude

společnost pronajímat majetek, jejím záměrem bude mít právě co nejvíce pochopitelné kategorie pro zákazníka. Jako příklady mohou být kategorie jednotlivých společností zaměřených na pronájem.

Prvním příkladem je firma RAMIRENT CZ, která na svých internetových stránkách rozděluje své nabízené produkty více méně podle velikosti. Kategorie má nastaveny takto:

- Velká mechanizace
- Vysokozdvižné vozíky
- Výtahy, zdvihací zařízení
- Malá mechanizace
- Ruční nářadí
- Zahradní technika
- Kontejnery, zařízení staveniště
- Lešení, stavební stojky, shozy na suť
- Příslušenství

Uvádím pouze základní rozdělení, každá z těchto kategorií má své podkategorie, které již řeší bližší požadavky. Na těchto kategoriích a vůbec na zařízení firmy na pozemní stavby je znát, že takovéto kategorie nemusí vyhovovat každé evidenci, pro zákazníka je ovšem stručná a přehledná. Kategorie jsou pojmenovány podle názvu strojů.

Společnosti, které půjčují, ale mohou mít ke kategorizaci majetku jiný přístup. Příkladem budiž kategorizace společnosti DEK u své půjčovny strojů a nářadí také na svých internetových stránkách, viz níže.

- Bourání, vrtání, nastřelování
- Řezání
- Broušení a hoblování
- Hutnění povrchů
- Zemní práce
- Lešení, bednění a pažení
- Plošiny, výtahy, jeřáby
- Zařízení staveniště

- Manipulace a přeprava materiálů
- Zpracování, čerpání stavebních směsí
- Úprava betonových povrchů
- Čištění, zpracování stavebního odpadu
- Měření a diagnostika
- Elektrocentrály, topidla, kompresory
- Čerpání a odčerpávání
- Nářadí pro izolatéry
- Nářadí pro klempíře a pokrývače
- Nářadí pro zedníky a malíře
- Zahradní technika
- Příslušenství

Oproti předchozí kategorizaci jsou tyto kategorie již v první úrovni obsáhlejší, konkrétnější. Největším rozdílem je ale přístup k rozřazení strojů a nářadí. Zákazník zde nehledá název stroje, ale hledá činnost, kterou bude realizovat a bude na ní potřebovat mechanizaci. Je tedy obecně řečeno uživatelsky přístupnější neodborné veřejnosti.

Ke zvolení správných kategorií je možno uvažovat ještě jiným způsobem. Při sestavování rozpočtů v programech zaměřených na tuto činnost přidává rozpočtář položky, které v sobě mají rozdělenou činnost v takzvaných TOV (technicko-organizační varianta) rozborech. V těchto rozborech je nejen činnost dělníka, materiál, ale také práce strojů podle kalkulačního vzorce. Každý stroj v databázi má i svůj vlastní kód, je tedy jasné, že samotné databáze mají kategorie, podle kterých jsou stroje rozřazeny. Tyto kategorie jsou k dispozici například v programu Callida, vycházející z ceníku ÚRS Praha, a jsou následující:

- Velká mechanizace
    - Rypadla, rýhovače a frézy
    - Dozery
    - Skrejpry
    - Grejdry
    - Traktory
    - Válce a hutnící stroje

    - Vrtné soupravy a beranidla
    - Protlačovací soupravy
    - Univerzální nosiče
    - Jeřáby
    - Výtahy, plošiny, vrátky
    - Finišery
    - Drtiče
    - Dokončovací stroje
    - Stroje na pokládku kolejí
    - Stroje pro zemědělství
    - Remorkéry, prámy, čluny
- Podzemní mechanizace
    - Razící štíty, těžní věže, nakladače
    - Důlní lokomotivy a vozy
- Dopravní mechanizace
    - Nákladní automobily
    - Domíchávače a čerpání betonu
    - Dopravníky
    - Stroje pro značení a čištění komunikací
    - Návěsy
    - Kolejová vozidla a vozíky
- Malá mechanizace
    - Míchačky, omítačky, injektážní soupravy
    - Vysokozdvižné vozíky
    - Elektrocentrály
    - Dávkovače a transportní sila
    - Kompresory, ventilátory, chladící věže
    - Napínací zařízení
    - Brusky, frézy
    - Ohýbačky
    - Svářečky
    - Zařízení na stříkání barev
    - Čerpadla
    - Svářečky plastů

- Hydraulické zvedáky
- Tesařské stroje
- Ostatní mechanizace
- Gama zářiče a detektory

Je zřejmé, že kategorie jsou poměrně obsáhlé. Vzhledem k tomu, že v rozpočtových programech jsou přiřazeny k jednotlivým položkovým pracím, a zároveň jsou rozpočtové programy v praxi hojně využívané, jedná se tedy pravděpodobně o fakticky nejlépe vyhovující kategorizaci ve stavebnictví. Navíc na rozdíl od půjčoven zaměřených na prostší a nespecifické práce podchycuje i specializované stroje a práce, jakými na příklad mohou být podzemní důlní stroje. Podobně jako u systému firmy RAMIRENT se u kategorií strojů v rozpočtech vyhledává název stroje nebo nářadí.

Pro účely této práce zvolím jako základní kategorie ty, které vychází z rozpočtových programů. Ovšem využití kategorií, jak je používá DEK, tedy vycházející z druhu činnosti, může být prospěšné při hledání konkrétního nářadí uživatelem. V praxi se jednotlivým nářadím slangově říká jinak, ku příkladu zedníci mají jinou slangovou terminologii než třeba horníci, byť se může jednat o stejné nářadí. Navíc se musí počítat i s tím, že hromadu prací ve stavebnictví realizují cizinci, tedy schopnost rozřadit mechanizaci podle více kritérií může být v důsledku příhodné.

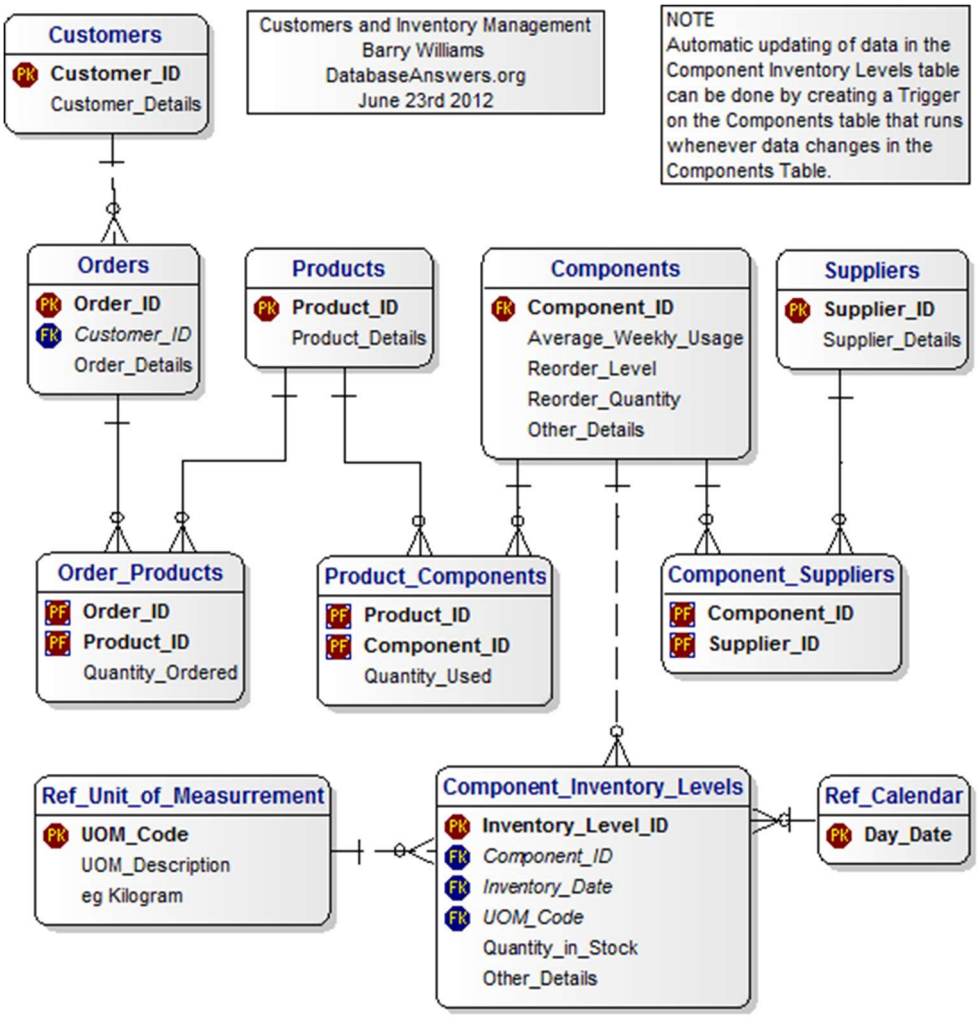
5 Návrh databáze

5.1 Příprava na návrh

Pro představu, jak takový model může vypadat, jsem se inspiroval modely na webu. Málokterý model ovšem odpovídá specifikům stavebnictví. Tyto modely mohou naznačit, jak postupovat při jednotlivých fázích modelování datové struktury.

Struktury jsou vyjádřeny v notaci Crow's Foot, která spočívá v znázornění entit v boxech a vztahy v obloucích s popisem. Na konci každého oblouku (čar spojující dvě entity) jsou symboly vyjadřující kardinalitu vztahu, čímž je na mysli počet možných vztahů.

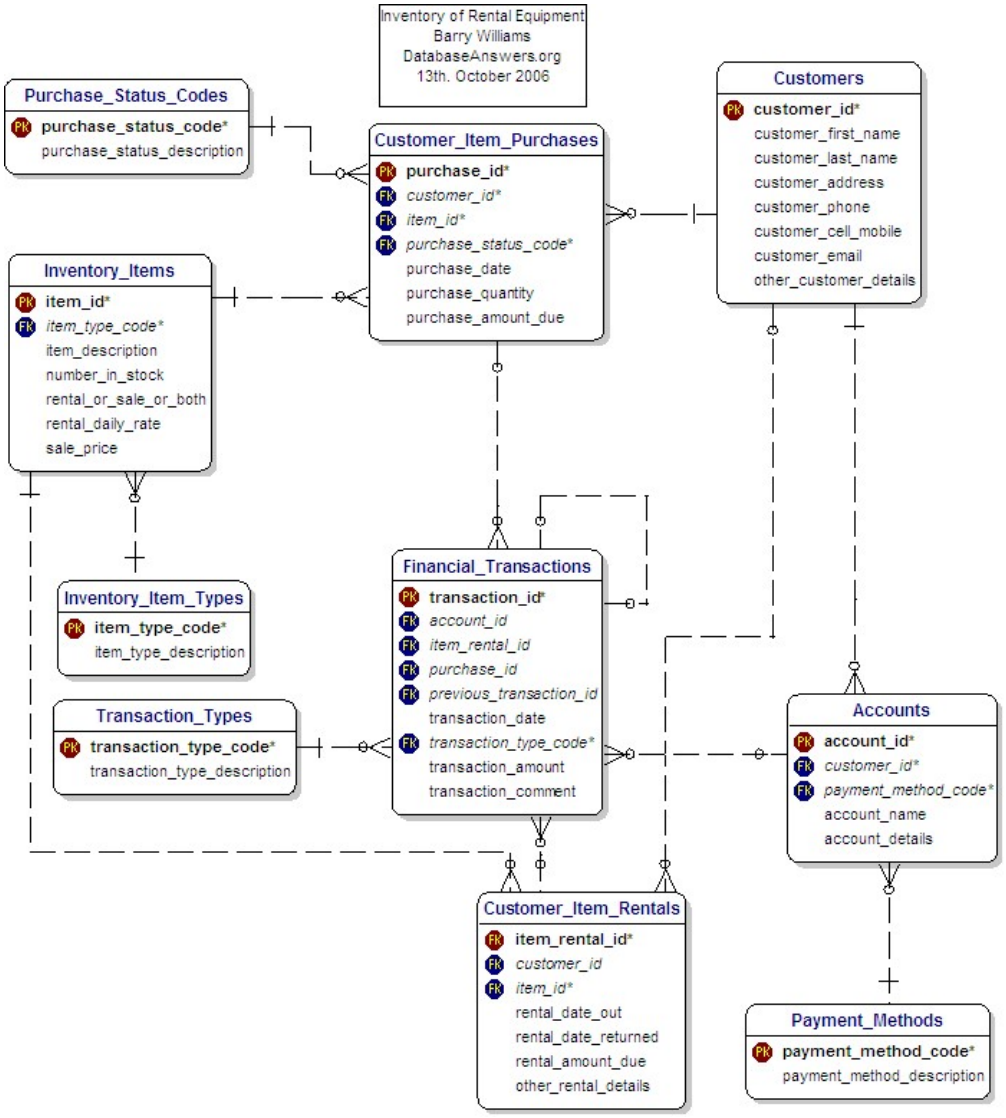
Prvním příkladem může být model pro zákazníky a řízení skladu.



Obrázek 9 - Model databáze zákazníků a skladu [5]

Tento model ukazuje, jak postavit databázi pro řízení skladu s komponenty, tvorbou produktů, řízením objednávek, dodavatelů a skladu v čase. Pokud uživatel potřebuje automatické obnovování databáze aktuálními daty, je zde potřeba vytvořit spouštěč, který bude obnovovat tabulku komponentů („Components“). Tento model můžeme využít, pokud prodáváme výrobek vytvořený z několika komponentů, příkladem ze stavebnictví může být ocelová konstrukce tvořená z komponentů typu I, HEB či jiných. V levém dolním rohu jsou v tomto modelu zakomponovány i informace o vlastnostech komponentu, kterými mohou být v českém stavebnictví normy EN ČSN či kódy certifikace ISO.

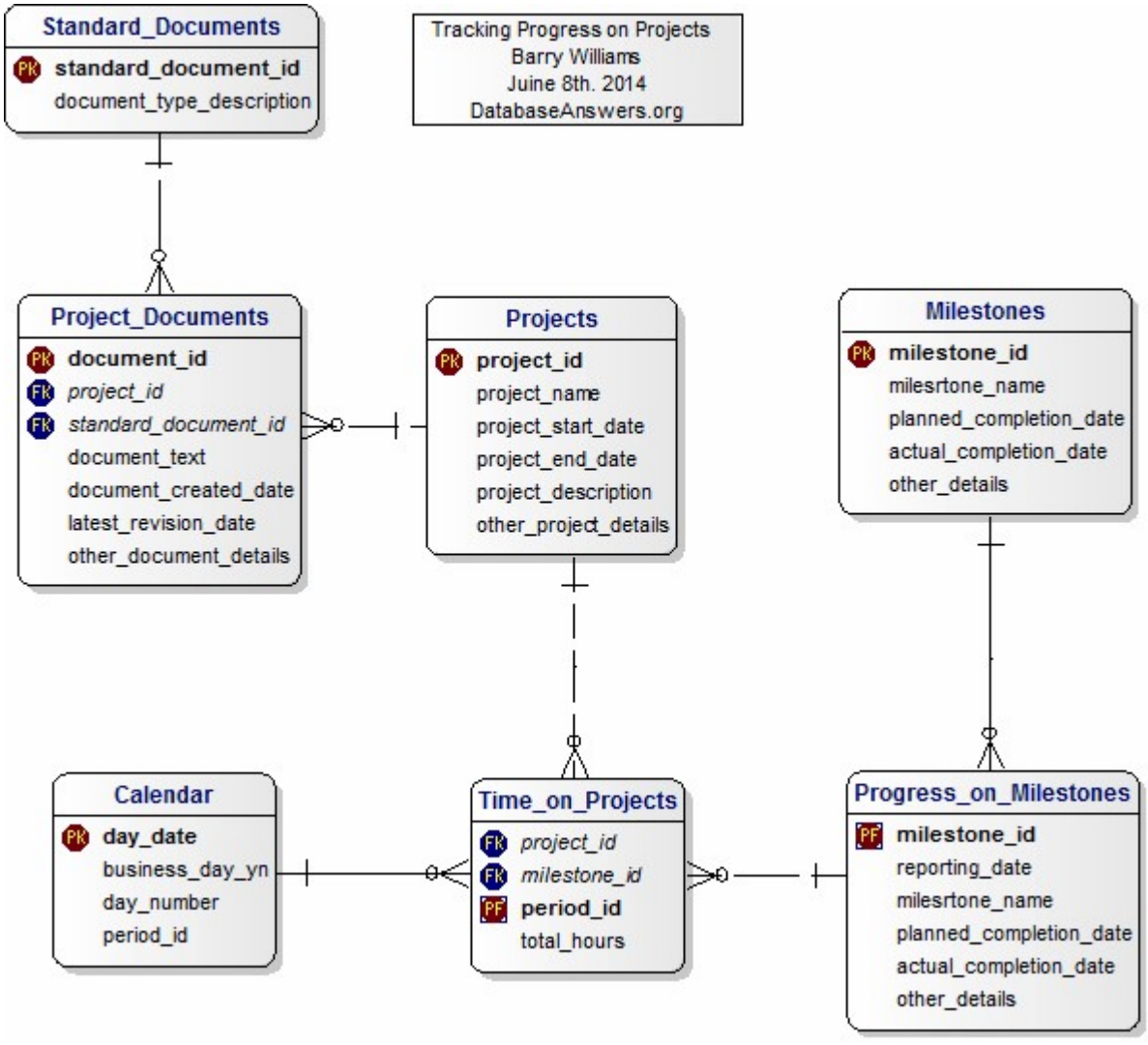
Dalším příkladným modelem budiž inventář půjčovny.



Obrázek 10 - Graf modelu databáze půjčovny [5]

Model dává možnost půjčit či prodat položky z inventáře, a zároveň je zde zakomponovaná možnost finančního vyrovnání, která je ve středu tohoto modelu. Vzhledem k tomu, že při vypůjčování jsou potřeba veškerá data o zákazníkovy, i tento model s tím počítá, a to v podobě tabulky v pravém horním rohu. Automatické obnovování databáze je řešeno spouštěčem při každé změně v tabulce „Financial_Transactions“.

Důležitá věc ve stavebnictví je plánování. I tento proces se lze řešit díky databázím. Ukázkový model sledování progresu projektů může vypadat takto.

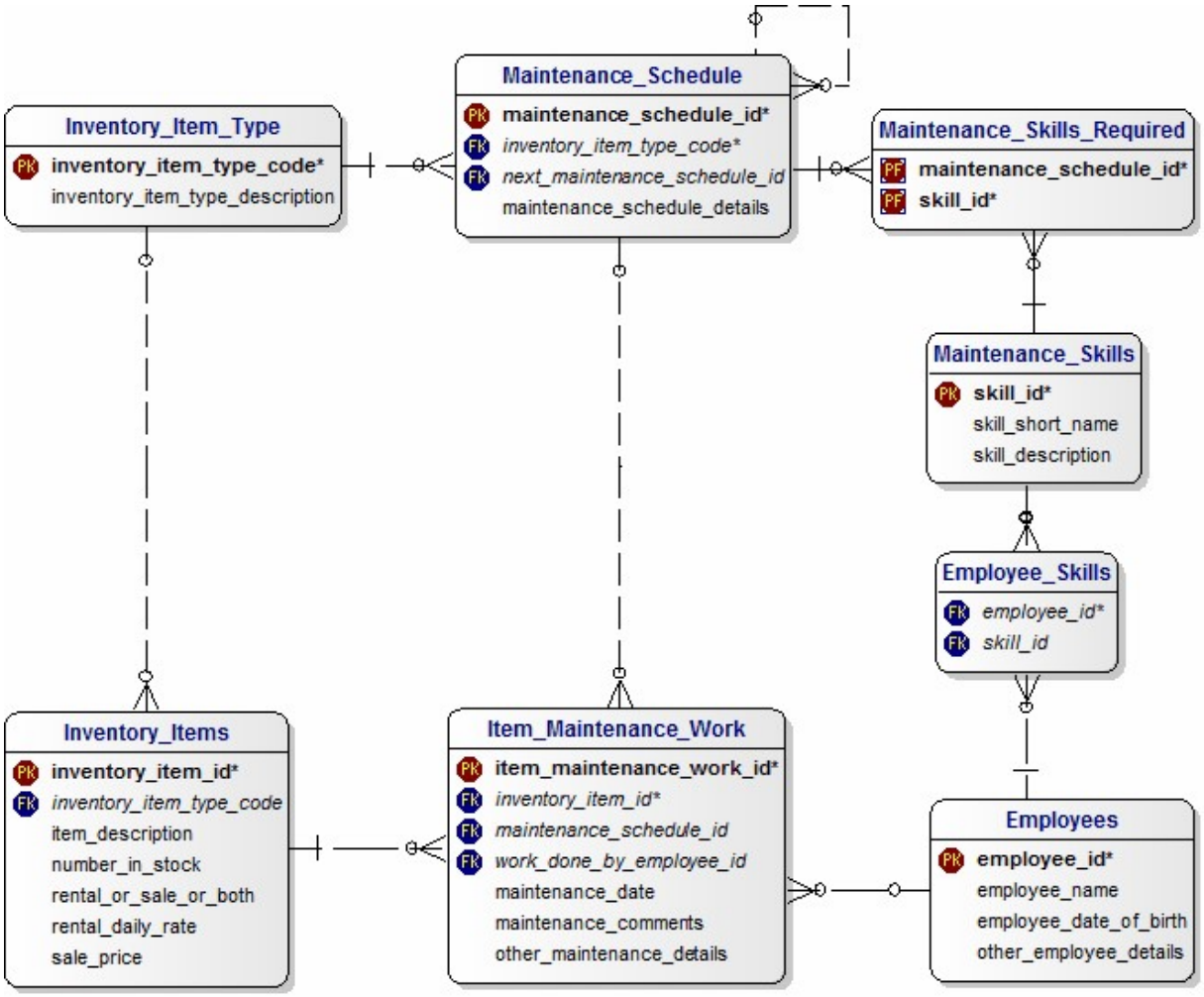


Obrázek 11 - Graf modelu databáze sledování průběhu realizace projektu [5]

Tento model sleduje progres projektu tak, že sleduje milníky, progres na nich a z toho plynoucí čas, který jednotlivé projekty vyžadují. Je to práce v čase, tudíž je potřeba přidat tabulku, která plní funkci kalendáře. Součástí modelu jsou informace o

projektu, jejich dokumentace a standartní dokumentace. Tato část týkající se dokumentace by měla pro stavebnictví vypadat tak, aby odpovídala požadavkům pro dokončení projektu. V této části databáze by měla být obsažena projektová dokumentace v takovém stavu, jak vyžaduje ČKA s ČKAIT.

Dalším modelem, který by mohl být vhodný pro řešení evidence majetku, je model údržby majetku.



Obrázek 12 - Graf modelu databáze údržby [5]

V tomto modelu se pracuje s plánem údržby, který je prováděn jednotlivými údržbovými procesy. Levá část modelu spravuje inventář a zařazuje jej, pravá část přiřazuje pracovníky a jejich schopnosti k požadavkům na údržbu. Tento model se také obnovuje podle potřeby díky spouštěči v tabulce „Maintenance_Schedule“.

5.2 Základní požadavky

Pro začátek si potřebujeme sepsat dokumenty, se kterými bude datová sada databáze pracovat. Jedná se o první krok tvorby databáze. Jsou to první entity, které mezi sebou budou pracovat v databázi.

Ta se zaměřuje na evidenci majetku. Majetek, inventář, bude první takovou entitou, ale je potřeba připojit další, kterými budou společnosti, projekty, lokality, projekty, stavby, dodavatelé, sklady a další. Tyto entity se v ER-diagramu s využitím Chenovy notace systematiky zobrazí jako obdélníky, symbolizující objekty. Vztahy mezi nimi jsou vyjádřeny kosočtverci a atribut je vyjádřen popisem v elipse.

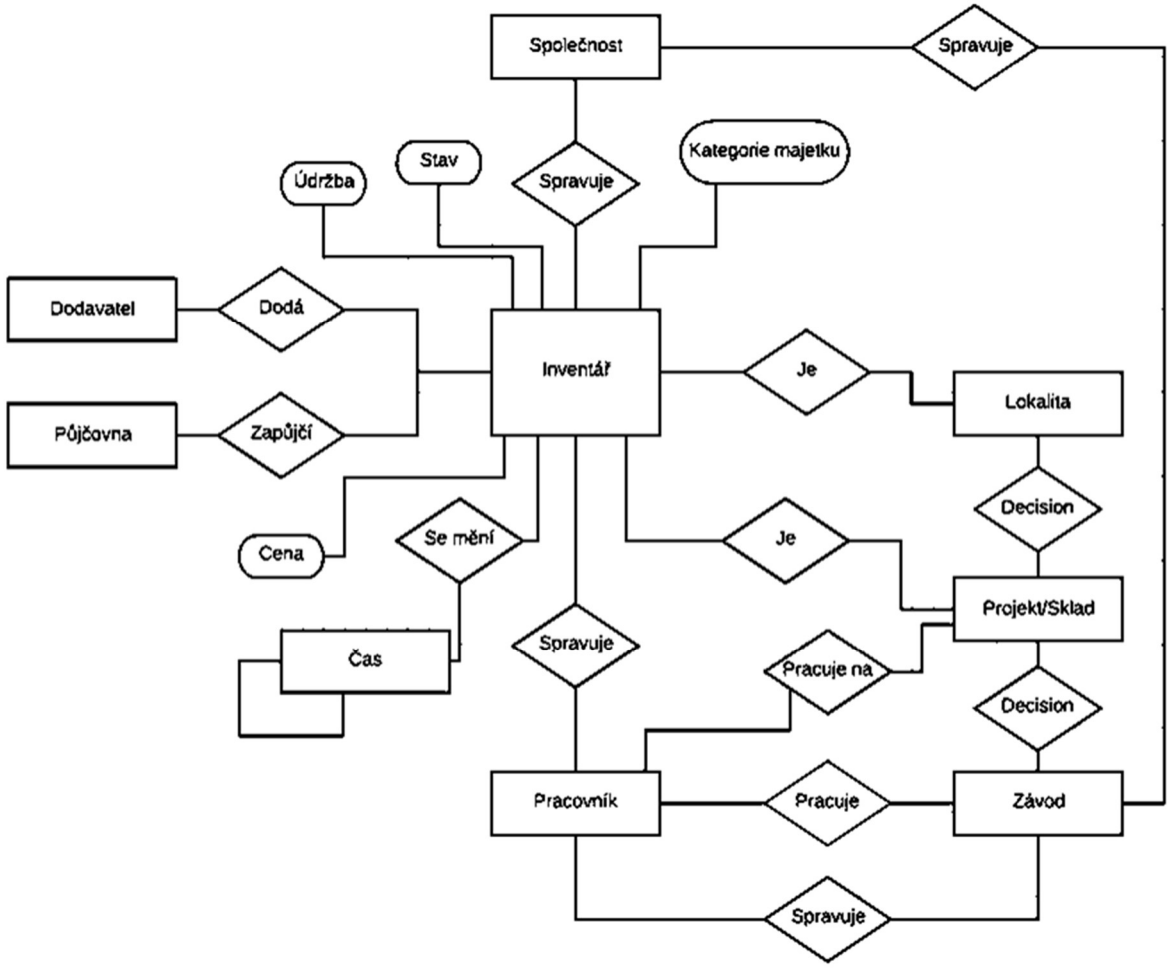
Tento návrh je v první fázi vývoje modelu databáze. Jedná se o takzvaný model reality neboli konceptuální návrh. Podstatou je potřeba teoreticky navrhnout databázi tak, aby odpovídala požadavkům.

Hlavní entitou evidence majetku je inventář. Ten se mění v čase. Každá položka v inventáři by měla mít přiřazenou lokalitu a projekt, na kterém se momentálně podílí. Další věcí důležitou pro fungování evidence je pracovník, který je zodpovědný za daný inventář. Tento pracovník je součástí závodu, tedy určité organizované části podniku, a je mu přiřazen projekt, na kterém by měl pracovat. Je třeba zmínit, že inventář je potřeba obměňovat, či využívat zapůjčení majetku, což je v diagramu řešeno pomocí dvou entit se vztahy nalevo od hlavního objektu inventáře. Celý systém je samozřejmě součástí celého podniku, který vlastní či spravuje majetek v inventáři. Celý tento systém návazností by měl fungovat a vzájemně se ovlivňovat během změn v procesu realizace projektů.

Inventář jako takový by mohl být navíc obohacen o atributy, které usnadní pozdější práci s analýzou dat dané databáze. Velkým tématem je vůbec zařazení majetku do kategorií. Toto téma již bylo předmětem této práce v kapitole "Kategorizace majetku", což vede ke snadnější práci s databází pro uživatele. Inventář by měl mít zároveň informace o stavu. Ideální pro analýzu dat řešit i využití jednotlivých položek majetku, jejich stav (v provozu, nevyužívaný, oprava) ovlivní návratnost investice do něj. S tímto stavem souvisí i údržba. Velmi složitým, ale neméně důležitým atributem inventáře, je cena. Metodika zjišťování hodnoty majetku je složité téma, účetní hodnota může být jiná než hodnota tržní, navíc se hodnota tohoto majetku v čase mění v závislosti na opotřebení, odpisování, trhu, měně a dalším proměnným. Je vhodné

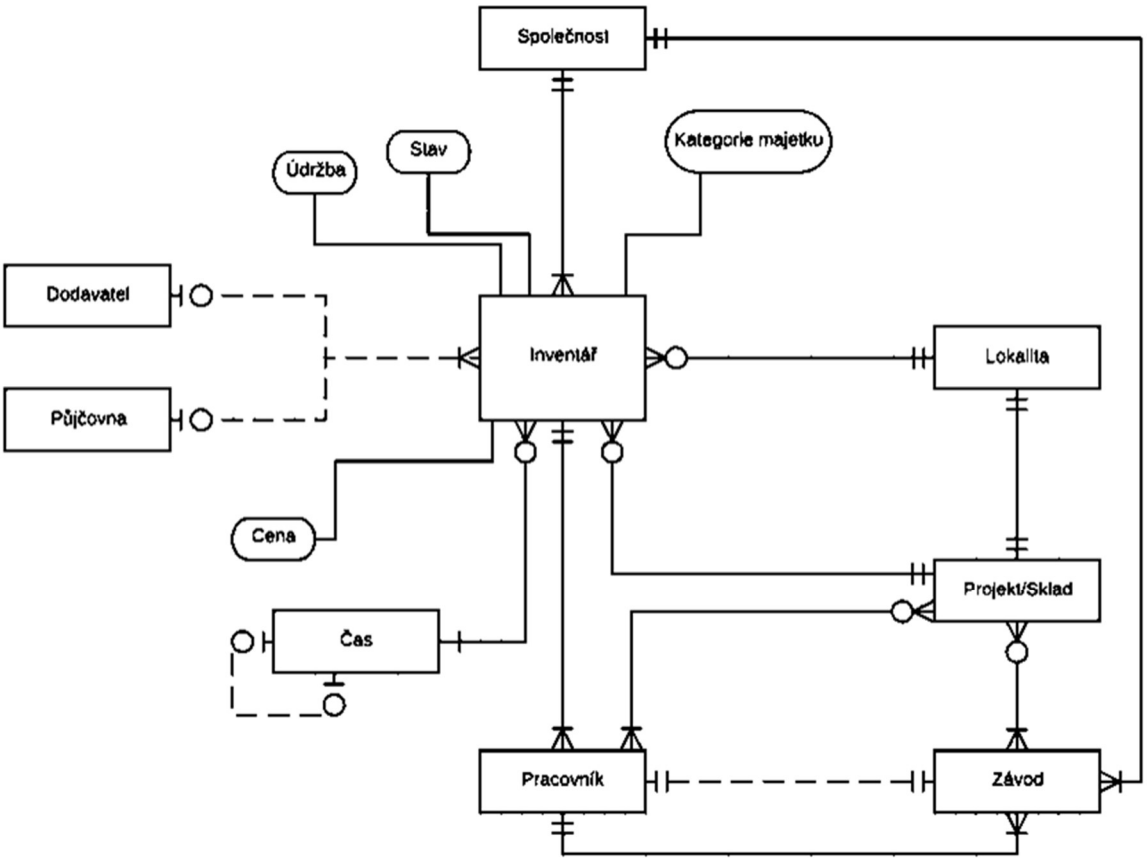
stanovit metodiku zjišťování ceny pro celou databázi a vytvořit spouštěč (trigger), který automaticky tuto cenu poupraví podle situace. Bylo by vhodné mít tyto vlastnosti zakomponované v databázi, ale technická řešení tohoto problému nepatří mezi jednoduchá.

Pro znázornění vztahů je nejlepším způsobem využití ER diagramu. Níže je představen diagram využívající Chenovu notaci. Pro realizaci tohoto diagramu bylo využito webové rozhraní pro tvorbu diagramů.



Obrázek 13 – ER model databáze správy majetku [vlastní výroba]

Pro lepší znázornění bude potřeba použít notace Crow's Foot. Tento druh ER digramu je takovým, ve kterém jsou znázorněny ukázkové databázové modely z předchozí kapitoly.



Obrázek 14 - Model databáze správy majetku metodou Crow's Foot [vlastní výroba]

V další fázi je potřeba doplnit atributy každému objektu v modelu. Tato fáze vývoje již bude spadat pod logický návrh. Vzhledem k tomu, že další fáze se již může realizovat v databázovém programu, v dalším kroku práce přejdeme k výběru softwaru.

6 Metodika práce

6.1 Vhodný software pro evidenci

V praxi nejvíce používaným systémem jsou tabulkové procesory, zejména MS Excel. Tento program je ovšem tabulkovým kalkulátorem a tvorba databáze není jeho primárním účelem. I tak ale může být nápomocen při tvorbě databází a při tvorbě výstupů z těchto databází. Vzhledem k tomu, že je Excel a jeho podobné programy určeny pro řešení široké škály úloh (nejen evidování), jsou tyto tabulkové procesory neefektivní pro řešení databází, jejíž tabulky mají mnohem více navazujících vlastností (relace, klíče a jiné). Excel se tedy využívá tehdy, když je potřeba něco vytvořit rychle a snadno. Pro složitější úlohy a velké objemy dat je ale nevhodný.

Trh nabízí celou škálu databázových programů a hromada jich je i dostupná volně. Nabízí to nesmírnou možnost vývoje databázových systémů uživateli. A vzhledem k tomu, že většina IT oddělení firem pracuje s minimálními rozpočty, volně dostupné databázové programy jsou vítanou možností pro vývojáře. Tyto nabízené programy budou níže představeny v abecedním pořadí. [6]

6.1.1 Cubrid manager

CUBRID je volně dostupný relačně-databázový manažerský systém s objektovým rozšířením v SQL jazyce, který byl vyvinut společností Naver Corporation pro webové rozhraní v roce 2008. Název se skládá z kombinace slov cube (kostka, přesněji zapečetěná krabička symbolizující zabezpečení) a bridge (most, který symbolizuje datové přemostění mezi relačními databázemi s nerelačními). Je považován za jednoho z nejlepších databázových programů na trhu, zejména z pohledu toho, že jeho grafické uživatelské rozhraní usnadňuje práci s developerskými jazyky jako jsou JDBC, PHP, Python, Pearl a Ruby. CUBRID zároveň nabízí nonstop online webový servis a zálohuje data online. Jeho nevýhodou je, že nemá debugger skriptů, nespolupracuje s Apple systémy a manuál je pouze v angličtině a korejštině.

6.1.2 Firebird

Dalším programem pro tvorbu SQL relačních databází je program Firebird, který může fungovat na Linuxu, Microsoft Windows, Mac OS X a dalších operačních systémech. Jeho devízou je plná podpora uložených procedur či spouštěčů (triggerů), což jsou objekty uložené v databázi bez jakýchkoliv dat, ale s programy, které poté

pracují s daty v databázi (mění je, využívá je…). Firebird plně podporuje transakce ACID a ku příkladu podporuje čtyři architektury (SuperClassic, Classic, SuperServer a Embedded). Komunita vývojářů používajících Firebird je velká a poskytuje volnou podporu. Je velmi specifický, což má za následek nedostatečnou kompatibilitu s jinými databázovými systémy.

6.1.3 MariaDB

MariaDB je velmi využívaný populární program pro vytváření databáze. Pracuje s tímto programem Wikipeda, Facebook, a dokonce i Google. Jedná se o program, který je vyvinut vývojáři, kteří pracovali na programu MySQL před tím, než byl odkoupen firmou Oracle. MariaDB navazuje na vývoj MySQL, ale pokračuje svojí open-source cestou. Prioritou tohoto databázového softwaru je bezpečnost, která, když je potřeba, je celou komunitou uživatelů MariaDB zlepšována formou patchů a updatů. Hlavními výhodami MariaDB jsou snadná rozšiřovatelnost, okamžitý přístup ve skutečném čase, základní kompatibilita s funkcemi MySQL a široká využívanost programu, který se de facto vyvíjí více než 20 let. Má nevýhodu v tom, že nemá moc praktických nástrojů pro urychlení reakce databáze.

6.1.4 MongoDB

Tento databázový program byl vyvinut v roce 2007 jako produkt „databáze pro velké nápady“. Byl vyvinut lidmi ze společností DoubleClick, ShopWiki nebo Gilt Groupe a zafinancován společnostmi jak Fidelity Investments, The Goldman Sachs Group, Inc. a Intel Capital, což už samo o sobě znamená, že to je projekt předurčen k úspěchu. MongoDB byl od svého uvedení na trh stažen více než 20 milionkrát a je podporován více než tisíci partnery, kteří předpokládají, že podporovaný produkt bude volně k dostání a kódování v něm bude jednoduché a přirozené. Velmi dobře pracuje s velkými daty (Big-data), je schopen poskytnout silnou komplexní analýzu dat, je schopen okamžité reakce i ve velkém objemu dat a jeho data mohou být transformovány do jakékoliv databáze. Na druhou stranu je práce s touto databází výzvou, neboť se velmi rychle vyvíjí, využívá poměrně složitý komplexní jazyk, což ztěžuje práci s ním.

6.1.5 MySQL

MySQL byl vyvinut v roce 1995 a je stále úspěšný na trhu. Nyní ve vlastnictví společnosti ORACLE mírně narušil svojí historii, která spočívala v tom, že byl průběžně vyvíjen komunitou uživatelů. MySQL je multiplatformní databáze, se kterou

se komunikuje pomocí jazyka SQL. Velmi oblíbená je kombinace s GNU/Linux, Apache, MySQL a PHP jako základní software webového servru. Vzhledem k tomu, že je MySQL standartním softwarem využívaným v průmyslu, je kompatibilní v zásadě s každým operačním systémem, který je napsán v programovacích jazycích C nebo C++. Je možné s ním pracovat i offline, flexibilní systém hesel a přístupů a vůbec zabezpečuje veškerá hesla v systému. Poté, co byl odkoupen společností ORACLE se zpomalil jeho vývoj, updaty jsou pomalé a komunita uživatelů již nemůže opravovat chyby či vyvíjet patche.

6.1.6 PostgreSQL

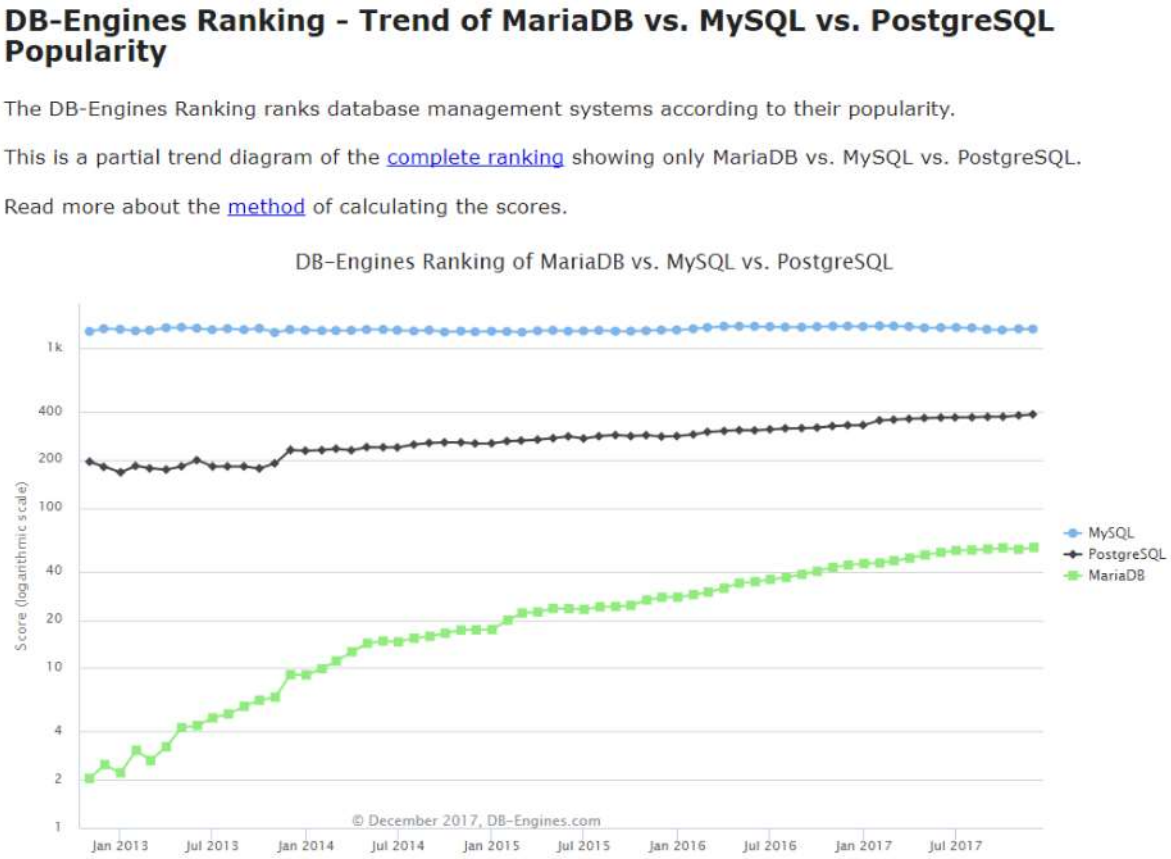
Tento program je vyvíjen více než 15 let, a jedná se o další open-source variantu tvorby databáze, která funguje na všech významných operačních systémech. Jedná se o objektově-relační databázový systém, který je vyvíjen pomocí globální komunity vývojářů a firem. V tom spočívá jeho největší výhoda, a tou je velmi rozsáhlá online podpora. Používá různé uživatelské datové typy a metody příkazů, je schopen pracovat v celé škále programovacích jazyků jakými jsou Java, Perl, Python, Ruby, Tcl, C/C++ a také ve vlastním pgSQL. Pokud je potřeba, PostgreSQL velmi dobře pracuje s geografickými a prostorovými daty. Snadná instalace je také jeho výhodou.

6.2 Analýza databázových programů

Analýzou jednotlivých výsledků zjišťuji, že nejvíce používané programy na trhu mají výhody v tom, že mají kvalitní online podporu, a to nejen poskytovanou správci a vývojáři, ale také komunitou uživatelů. Další jednoznačnou výhodou je, že jsou kompatibilní s nejvíce využívanými operačními systémy a zároveň s konkurenčními databázemi na trhu. Tato skutečnost dává uživateli těchto programů volnou ruku ve správě těchto systémů. Připuštěním těchto faktů se z výběru vypadávají CUBRID (nepodporován na Apple systémech) a Firebird (nedostatečná kompatibilita s jinými databázemi).

Složitost programu MongoDB také není vhodná pro řešení evidence, neboť se nejedná o úlohu, která by měla velký objem dat. I vzhledem k tomu, že já jako uživatel uvítám jednoduchost databázového systému, vybíral bych si z programů MariaDB, MySQL a PostgreSQL. Nejpopulárnějším programem je MySQL i přes to, že jej dotahuje PostgreSQL. MariaDB jakožto program vyvinutý z MySQL si od svého

uvedení na trh získává stále větší popularitu, stále však je používán pouze v polovině případů oproti originálnímu MySQL.



Obrázek 15 - Graf popularity databázových systémů [7]

Porovnával jsem tyto tři různé systémy řízení báze dat. Vzhledem k popularitě MySQL a zároveň tomu, že již s tímto databázovým systémem jsem mírně seznámen, budu pokračovat práci v MySQL.

Tabulka 2 - Tabulka pro porovnání databázových systémů [7]

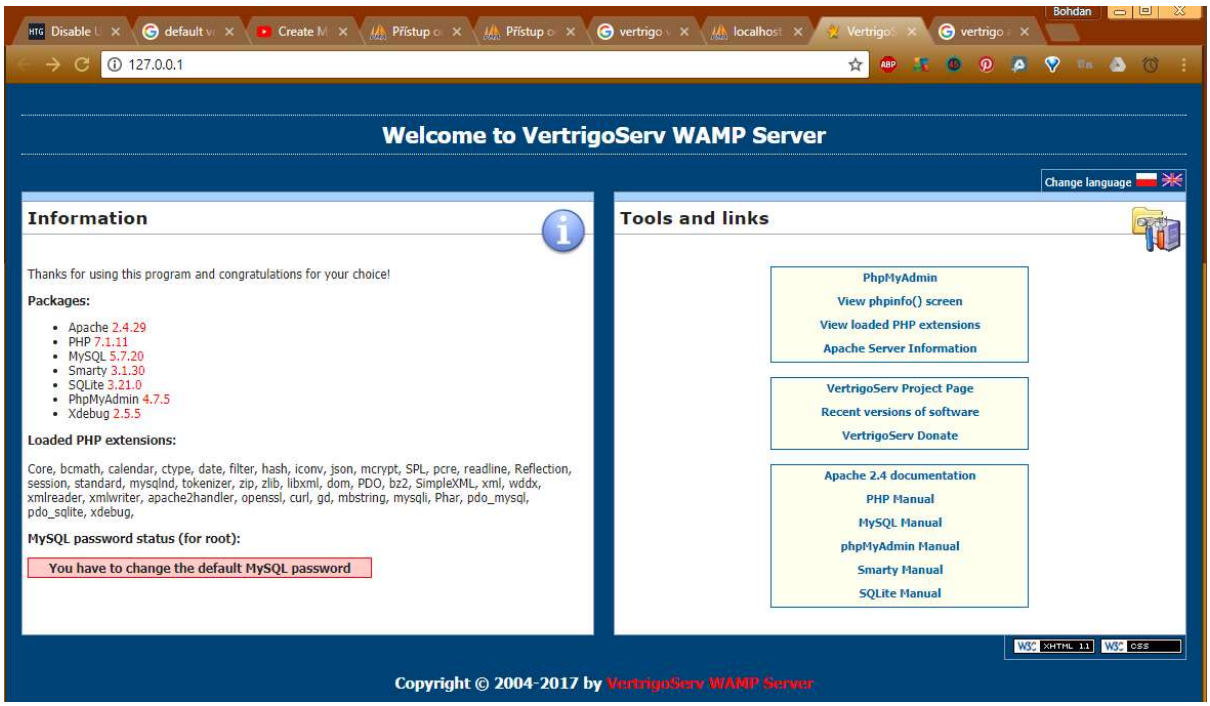
Jméno SŘBD	MariaDB	MySQL	PostgreSQL
Popis	Relační DBMS s volně přístupným licenčním kódem	Relační DBMS s volně přístupným licenčním kódem	Relační DBMS s volně přístupným licenčním kódem
Primární databázový model	Relační DBMS	Relační DBMS	Relační DBMS
Doplňující databázové modely	Uložiště dokumentů Úložiště klíčových hodnot	Uložiště dokumentů Úložiště klíčových hodnot	Uložiště dokumentů Úložiště klíčových hodnot
Žebříček popularity	**Skóre: 56.73** **Pořadí: #17 Celkově** **#10 Relační DB**	**Skóre: 1318.07** **Pořadí: #2 Celkově** **#2 Relační DB**	**Skóre: 385.43** **Pořadí: #4 Celkově** **#4 Relační DB**
Internetová stránka	mariadb.com mariadb.org	www.mysql.com	www.postgresql.org
Technická dokumentace	mariadb.com/kb/en/library	dev.mysql.com/doc	postgresql.org/docs/manuals
Vývojář	MariaDB Corporation Ab (MariaDB Enterprise), MariaDB Foundation (community MariaDB Server)	Oracle	PostgreSQL Global Development Group
Uvedení na trh	2009	1995	1989
Současná verze	10.2.11, Listopad 2017	5.7.20, Říjen 2017	10.1, Listopad 2017
License info	Open Source	Open Source	Open Source
Cloud-based info	ne	ne	ne
Implementační jazyk	C and C++	C and C++	C
Operační systémy serveru	FreeBSD Linux Solaris Windows	FreeBSD Linux OS X Solaris Windows	FreeBSD HP-UX Linux NetBSD OpenBSD OS X Solaris Unix Windows
Datové schéma	ano	ano	ano
Ruční psaní	ano	ano	ano
Podpora XML	ano	ano	ano
Druhotné indexy	ano	ano	ano
SQL	ano	ano	ano
APIs and other access methods	ADO.NET JDBC ODBC	ADO.NET JDBC ODBC	native C library API for large objects ADO.NET JDBC ODBC
Supported programming languages	Ada C C# C++ D	Ada C C# C++ D	.Net C C++ Delphi Java

	Eiffel Erlang Go Haskell Java JavaScript (Node.js) Objective-C OCaml Perl PHP Python Ruby Scheme Tcl	Delphi Eiffel Erlang Haskell Java JavaScript (Node.js) Objective-C OCaml Perl PHP Python Ruby Scheme Tcl	Perl PHP Python Tcl
Skripty na straně sereveru	ano	ano	Uživatelem definované
Triggery	ano	ano	ano
Metoda rozdělení	Horizontální	Horizontální	Deklarativní
Metody replikace	Master-master replikace Master-slave replikace	Master-master replikace Master-slave replikace	Master-slave replikace
MapReduce	ne	ne	ne
Koncept konzistence		Immediate Consistency	Immediate Consistency
Cizí klíče	ano	ano	ano
Koncept transakce	ACID	ACID	ACID
Řízení souběžnosti	ano	ano	ano
Podpora trvalosti dat	ano	ano	ano
Podpora procesů v paměti	ano	ano	ne

7 Konktrétní evidence – ukázka prototypu

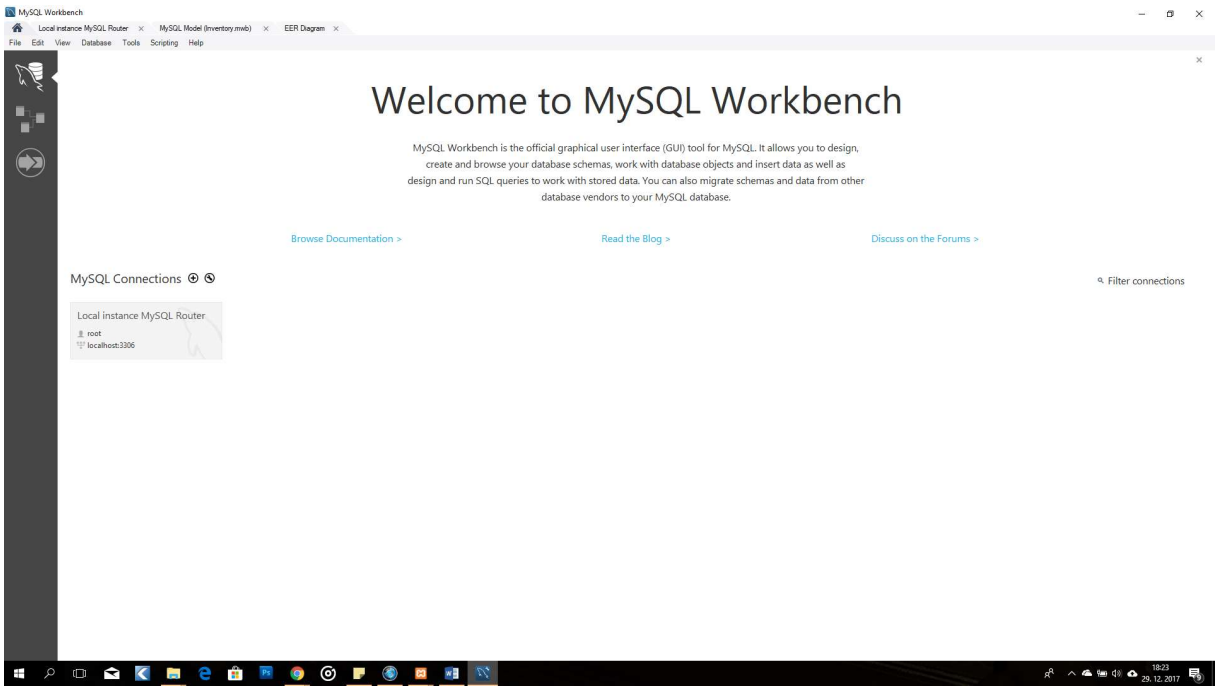
7.1 Vytvoření databáze

Pro ukázku prototypu si musím vytvořit lokální webový server pro vývoj a testování. Pro tuto potřebu se perfektně hodí softwarový balíček Vertrigo. Vzhledem k tomu, že většina webových serverů využívá stejné komponenty jako má v sobě zabudovaný Vertrigo, je pro případný přechod z testovací fáze na živý server velmi snadný.



Obrázek 16 - Screenshot služby VertrigoServ [8]

Další fází vytvoření samotného modelu v MySQL Workbench programu. Tento program v sobě zahrnuje možnost tvorby, designu a správy databáze, a to v poměrně vhodném grafickém zpracování. Vytvořením lokálního serveru můžeme tento program využít.



Obrázek 17 - Ukázka programu MySQL Workbench [1]

Po nastavení přístupových bodů začíná práce s modelem databáze podle vzoru ER diagramů, které jsou popsány a znázorněny v předchozích kapitolách.

Ústřední entitou je samotný inventář, který se nachází ve středu modelu. Tato entita má mnoho atributů. Primárním atributem je index položky, který musí být nezaměnitelný. Celkově má tabulka „Inventář“ tyto atributy:

- Index (Primární klíč)
- Název položky – nesmí to být nulová hodnota
- Kód položky (vnitropodnikový kód nebo čárový kód) – Tato hodnota se může opakovat, a to zejména v případě spotřebního materiálu, kterému není potřeba přidělovat vnitropodnikový kód.
- Popis položky – popis, ale také se v tomto bodě mohou nalézat slangové názvy určitého nástroje.
- Obrázek – Je vhodné přidat každé položce i vizualizaci pro usnadnění popisu.
- Index lokality – Tento index udá informaci o místě, kde se daný majetek nachází.
- Index zaměstnance – Jedná se o kód zaměstnance, který v daný čas má nástroj ve správě.

- Index společnosti – Tento index udává majitele či správce majetku v daném momentě.
- Index závodu – Tento index udává závod (divizi, department, sklad), který momentálně spravuje tento majetek.
- Index projektu/skladu – Tento index udává projekt, na kterém je momentálně nástroj využíván, nebo zdali je uskladněn. Jestli tato informace bude archivována v čase, může se využitím těchto dat zjistit vytíženost majetku.
- Index kategorie majetku – Pro lepší zařazení a správu majetku je zaveden tento index, jehož podstata je blíže popsána v předchozích kapitolách.
- Index údržby – Tato informace může pomoci v plánování nejen vytíženosti, ale i údržby daného majetku.
- Index stavu – Tato informace udává, zdali je majetek v užívání, v opravě či na odpis.
- Index dodavatele – Je vhodné evidovat dodavatele majetku pro případ reklamačních závad či jiných příhod.
- Index pronájmu – Je potřebné evidovat pronájemce, sledovat ceny a porovnávat jejich výhodnost.

Většina informací v této tabulce vychází z jiných tabulek a je reprezentována pouze indexy. Tento postup má za následek to, že se urychlí proces získávání a ukládání informací do databáze. Kdyby se vypisovaly veškeré informace z dané databáze, výrazně by to zpomalilo průběh práce s databází.

Další tabulky v databázovém modelu již nejsou tolik obsáhlé, a spíše rozšiřují základní informace inventáře, který je logickým středem zájmu v této evidenci majetku.

První tabulkou je „Společnost“. Udává základní informace o společnostech, které pracují s inventářem společnosti. Může se jednat o hlavní podnik, dceřiné společnosti, ale také společnosti, které spolupracují na určitých projektech či si určitý majetek/stroj půjčí pro realizaci svého zájmu. Pokud bude mít společnost zájem, může být v této databázi obsažena informace o společnostech poskytujících úvěr či leasing. V mém prototypu má tato tabulka tyto atributy:

- Index společnosti (primární klíč)
- Název společnosti
- Adresa společnosti
- Další informace o společnosti

Velké procento společností má rozdělenou svoji strukturu na divize, závody či jiné struktury, a to za účelem efektivnějšího vedení společnosti. Je vhodné evidovat majetek i v rámci těchto struktur ve společnosti. V mém prototypu má tato tabulka tyto hodnoty atributů:

- Index závodu (primární klíč)
- Název závodu
- Popis závodu
- Zaměstnance
- Společnost, pod kterou závod spadá

Další důležitou entitou ve společnosti jsou zaměstnanci. Je asi zbytečné upozornit na to, že jsou to v důsledku oni, kteří plní své pracovní povinnosti za pomocí majetku v inventáři. Tabulka, která je eviduje, vypadá v mém prototypu takto:

- ID Pracovníka (primární klíč)
- Jméno
- Příjmení
- Závod, pod který spadají
- Společnost, pod který spadají
- Telefonní číslo
- Číslo na mobil
- E-mailový kontakt
- Adresa
- Další informace

Společnost má projekty, které realizuje, a sklady, na kterém má uskladněný materiál, stroje, či jiný majetek, který pro případné řešení projektů využívá. Je vhodné , aby společnost měla tyto projekty rozdělené, zejména z pohledu sledování nákladů a dalších finančních toků. V prototypu tabulka projektů a skladů vypadá takto:

- Index projektu nebo skladu (primární klíč)
- Název projektu či skladu
- Počátek projektu
- Zakončení projektu
- Popis projektu nebo skladu
- Ostatní detaily projektu či skladu
- Index lokality

Důležitou informací v evidenci majetku je lokalita. Tu je vhodné sledovat pro případné zamezení ztráty majetku, a zároveň je vhodné sledovat pohyby jednotlivých položek. S majetkem se přesunují i náklady s ním spojené. Sledování těchto pohybů v čase a pozdějším výpočtem je možné spočítat náklady na přesun, a to různými způsoby. Tím nejjednodušším by mohlo být využití geografické polohy (zemská výška a šířka) dvou bodů, a pomocí rotačních souřadnic a poloměru Země zjistit jejich vzdálenost od sebe. Tento postup je možné využít ve skriptu, který by poté mohl vyhodnotit celkové přesuny majetku. Proto jsem mimo jiné do tohoto prototypu přidal i souřadnice zemské výšky a šířky. Celkově by tabulka vypadala takto:

- Index lokality
- Název lokality
- Adresa lokality
- Zemská výška
- Zemská šířka

Další tabulkou je kategorie majetku. Tato kategorizace je již popsána v jedné z předchozích kapitol a její účel je zejména logické zařazení majetku do kategorií tak, aby uživatel, ať jím mám na mysli kohokoliv od manažera po dělníka, měl práci s danou databází co nejjednodušší. Tabulka v prototypu vypadá takto:

- Index kategorie (Primární klíč)
- Název kategorie
- Popis kategorie

Další tabulkou, která je řešená v prototypu, jest údržba. Informace o ní může být přínosná v součinností s aktuálním srovnáním stavu majetku. Plán údržby se v některých případech realizuje v souladu s nařízeními a zákony (příkladem budiž

technická kontrola automobilů), v některých případech se realizuje prakticky (způsob až se to porouchá, tak se to opraví). Je ovšem vhodné údržbu nepodceňovat nejen z bezpečnostních důvodů, ale i důvodů záručních či z důvodu pozdějšího plnění pojištění. Evidence údržby je tedy vhodná a ekonomicky výhodná. V prototypu vypadá takto:

- Index údržby (primární klíč)
- Index majetku
- Název majetku
- Plán příští údržby
- Detaily údržby

Další tabulkou je tabulka stavu majetku. Je potřeba mít přehled o majetku, jestli je v užívání, nevyužíván, v opravě či na odpis. V prototypu vypadají atributy této tabulky takto:

- Index stavu majetku
- Popis stavu majetku

Poslední dvě tabulky jsou velmi podobné. Jedná se o tabulky dodavatelů a pronajímatelů položek majetku. Jedná se o tabulky, ve kterých je zejména jejich popis. V prototypu vypadají takto:

- Dodavatel
    - Index dodavatele
    - Název dodavatele
    - Detail dodavatele
    - Adresa dodavatele
    - Telefon dodavatele
    - Mobilní telefon dodavatele
- Půjčovna
    - Index půjčovny
    - Název půjčovny
    - Detail půjčovny
    - Adresa půjčovny
    - Telefon půjčovny

- Mobilní telefon půjčovny

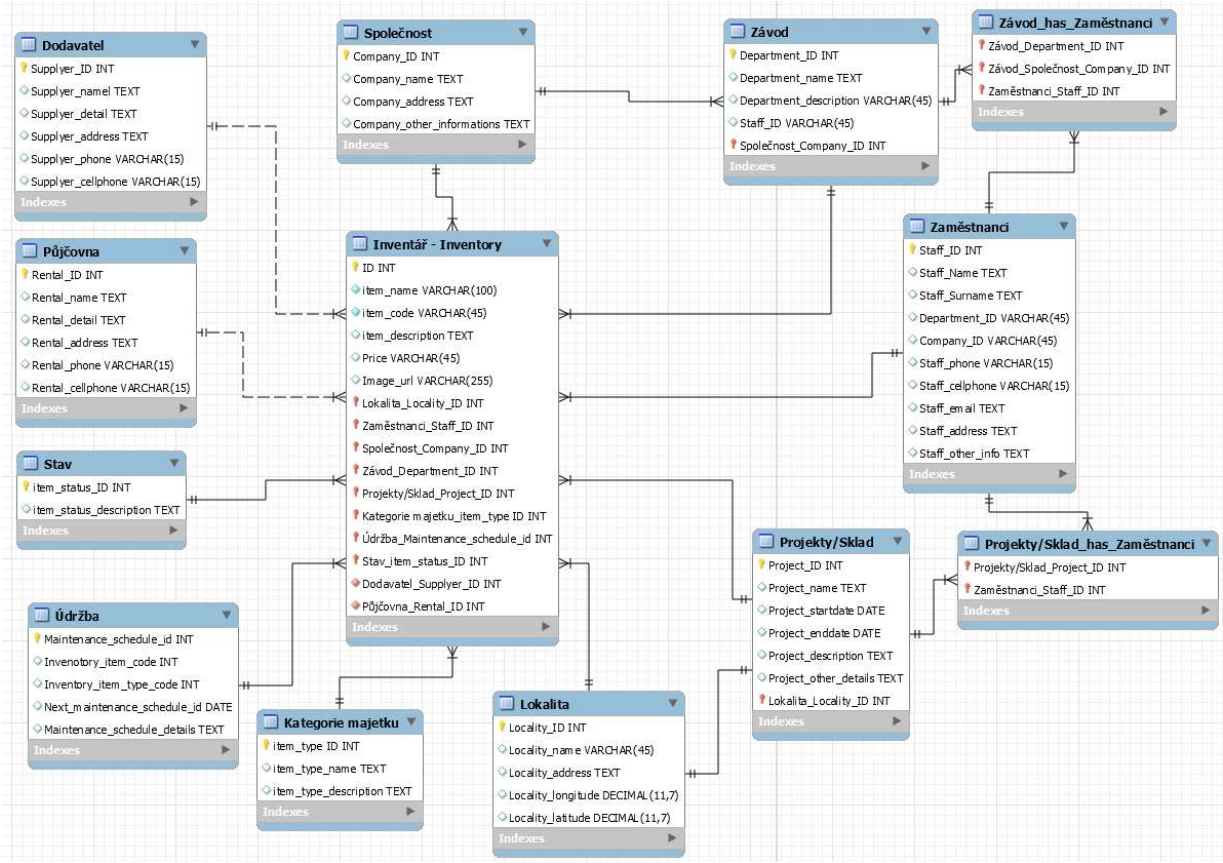
V ER-modelu, který byl zpracován v předchozích kapitolách, jsou řešeny návaznosti jednotlivých entit. Přidáním těchto entit a jejich návazností musíme vytvořit přídavné tabulky, které řeší vztahy M:N v databázi. Tyto vztahy jsou v prototypu dvě a tyto tabulky v prototypu vypadají takto:

1. „Závod“ má „Zaměstnance“, jeden závod má více zaměstnanců, ale zároveň zaměstnanec může pracovat na agendě více závodů.
    - Index závodu
    - Index zaměstnance
    - Index společnosti (pro zvláštní případy)
2. „Projekt“ má „Zaměstnance“, na jednom projektu pracuje více zaměstnanců, ale zároveň může zaměstnanec pracovat na více projektech naráz.
    - Index projektu
    - Index zaměstnance

Pokud se tyto tabulky zakomponují do databázového modelu v programu MySQL Workbench, jsou zachována jejich propojení.

Ke každému atributu musí být přiřazen datatyp, který odpovídá podstatě atributu. Pokud se jedná o název, popis či třeba adresu, vhodným datatypem je text. Datum zahájení a datum konce musí být v datatypu datumu. Číslo jako číslo s definovaným počtem číslic a podobně.

Realizované schéma prototypu databáze vypadá takto:

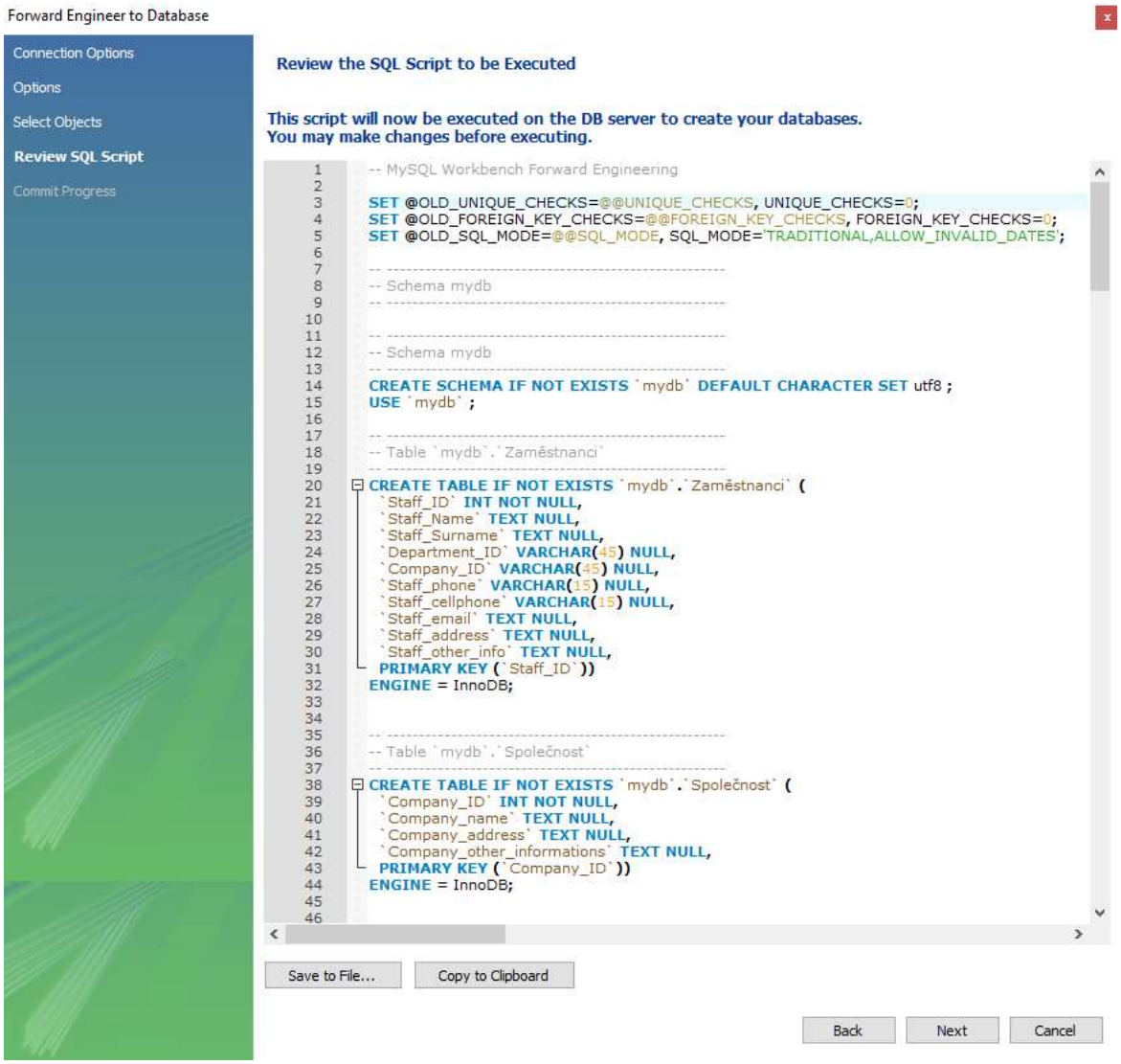


Obrázek 18 - Vlastní zpracování modelu databáze v MySQL workbench [1]

Realizovaná databázový model je uložen ve formátu *.mwb, což je formát programu MySql Workbench pro ukládání databázového modelu.

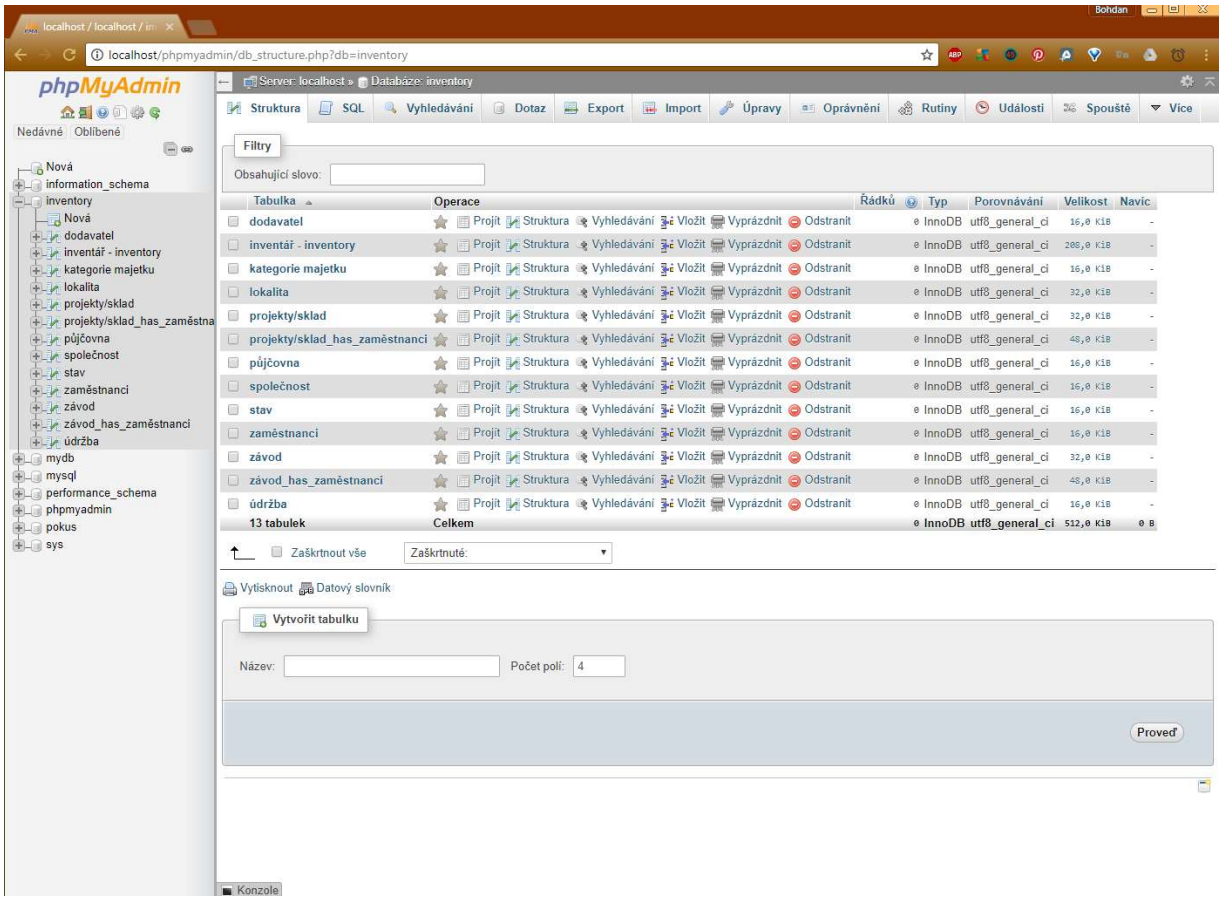
Podstatou dalšího kroku bude přesun databáze do lokálního serveru pomocí služby phpMyAdmin. Ta je v balíčku Vertrigo, která poskytuje lokální server a další nástroje pro testování databáze.

Databázový model musím vytvořit v jazyce SQL, a to lze udělat snadno a rychle pomocí možnosti „Forward Engineer to Database“ v MySql Workbench. Po několika kliknutí tento postup vytvoří kód, který vytvoří jednotlivé tabulky databáze. Výsledný proces tvorby databáze vypadá takto:



Obrázek 19 - Generovaný SQL skript pro vytvoření databáze podle modelu [1]

Jak je možné vidět, program vytvořil skript, který vytvoří celou databázi tak, jak byla vymodelována v datovém modelu programu MySQL Workbench. Tato databáze se vytvoří na lokálním serveru počítače a je možné jí testovat a spravovat ve webovém rozhraní programu phpMyAdmin.



Obrázek 20 - Ukázka databáze ve webovém rozhraní nástroje phpMyAdmin [9]

7.2 Vložení dat

Vzhledem k tomu, že nejsou k dispozici žádná konkrétní data, se kterými bych mohl pracovat, musím si vytvořit modelový příklad pro testování této databáze.

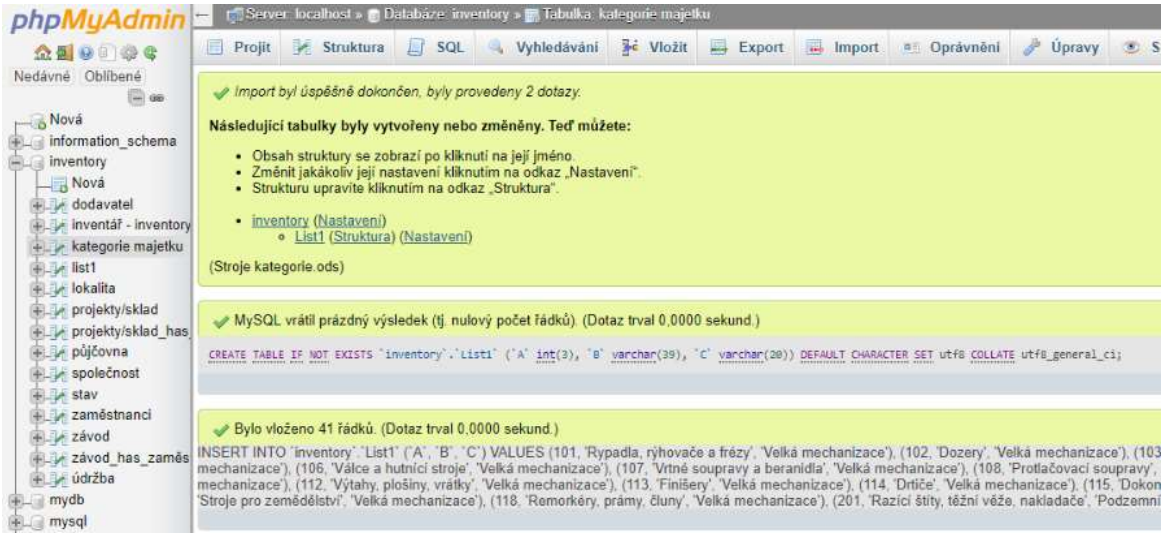
Zvolil jsem si tedy společnost, která vlastnit mechanizaci řešenou v této databázi, přiřadil jí jednotlivé závody, přidal projekty, zaměstnance a lokality. Stavy majetku jsem rozdělil na stavy:

- V užívání
- Nevyužíván
- Vyžaduje opravu / v opravě
- Na odpis

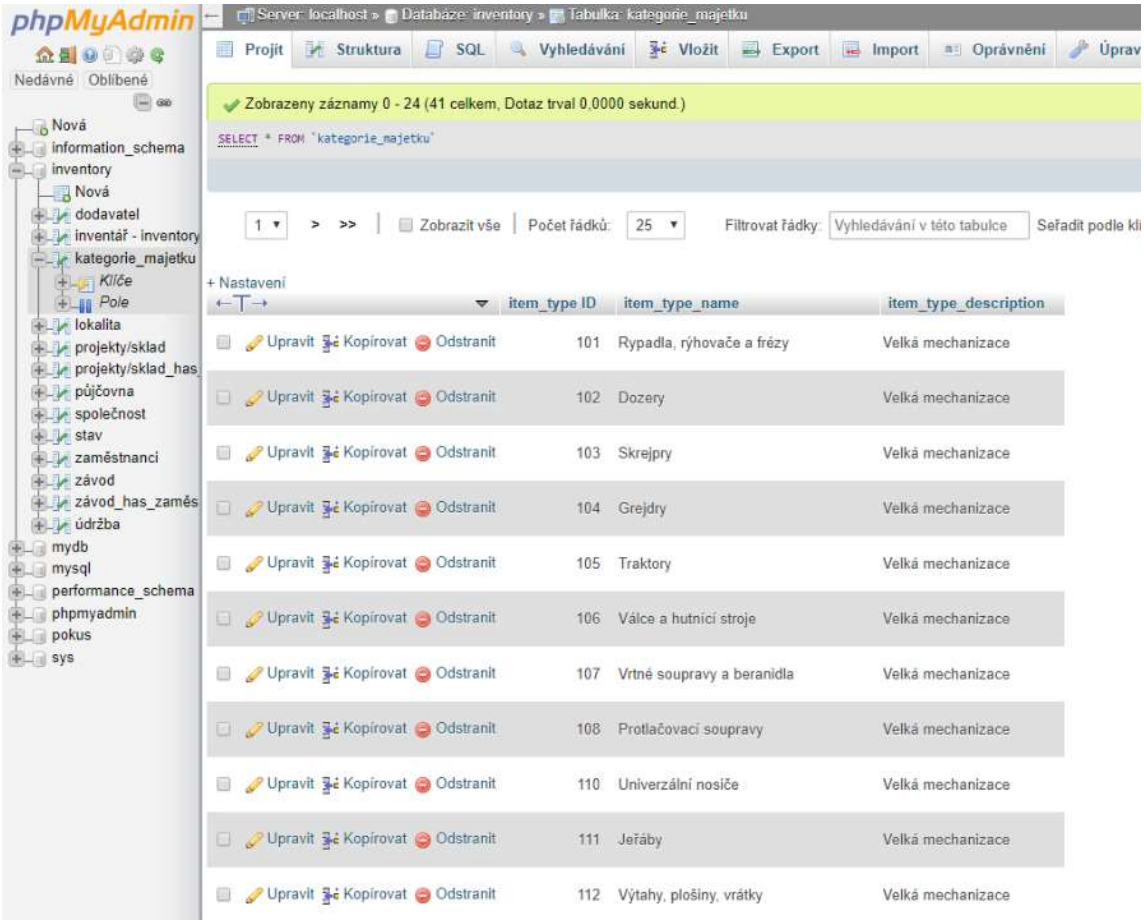
Dalšími tabulkami jsou vytvořené tabulky údržby, dodavatelů a pronájmů.

Jedinou tabulkou, která byla importována z externího zdroje, je tabulka kategorií majetku. Ta kategorizace, jejíž výběr byl popsán v předchozích kapitolách, byla převedena do tabulky v excelu a ve formátu „OpenDocument Spreadsheet“ (*.ods)

importována do databáze na lokálním disku. Screenshoty z importu jsou obsaženy níže.



Obrázek 21 - Import dat ve webovém rozhraní phpMyAdmin [9]

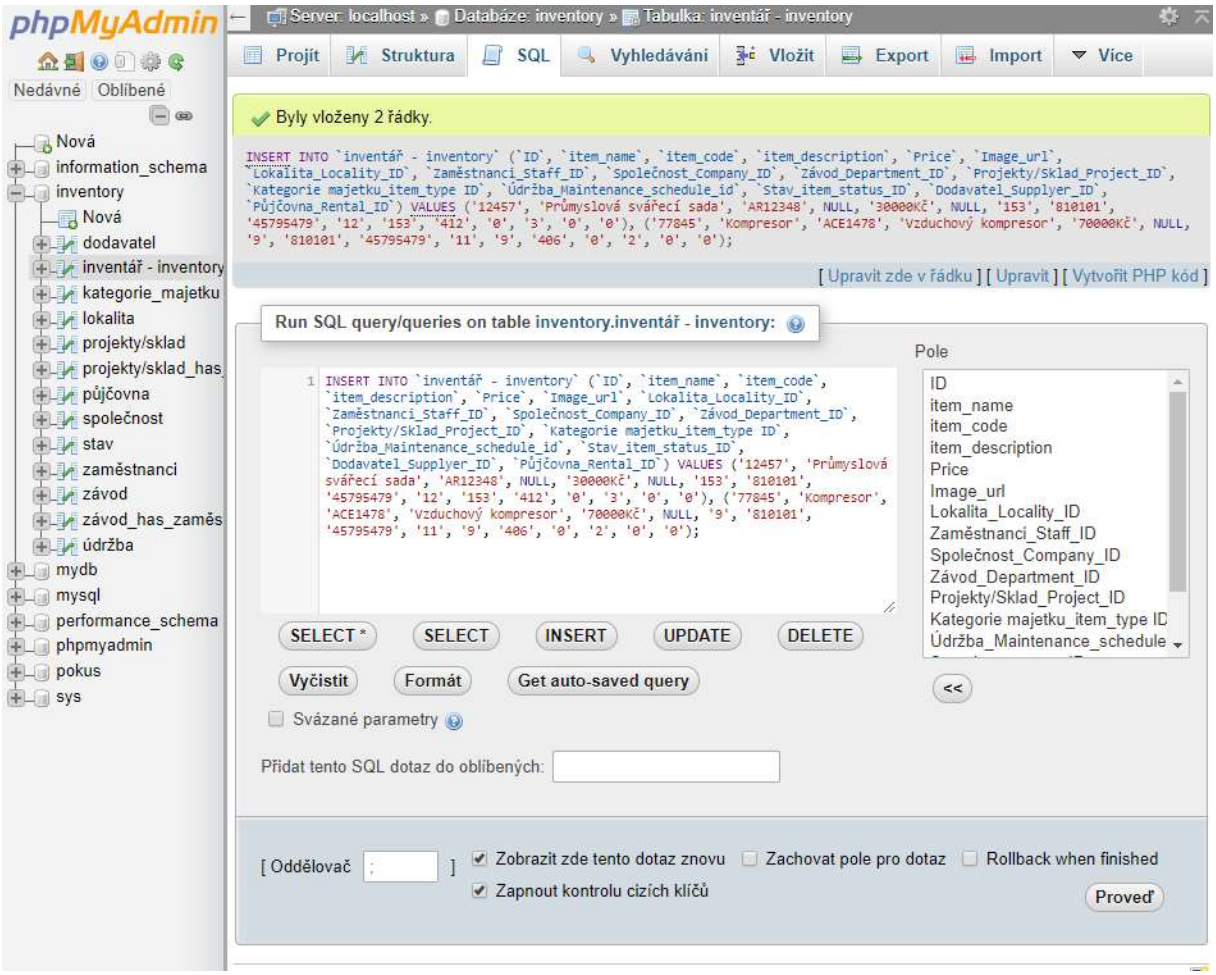


Obrázek 22 - Importovaná data zobrazená v databázi, kategorie majetku [9]

Podstatou databáze je inventář mechanizace. Do této tabulky jsem přidával položky v posledním kroku, a to zejména z toho důvodu, že samotná tabulka „inventář“ obsahuje mnoho cizích klíčů neboli indexů, které vychází z jiných tabulek.

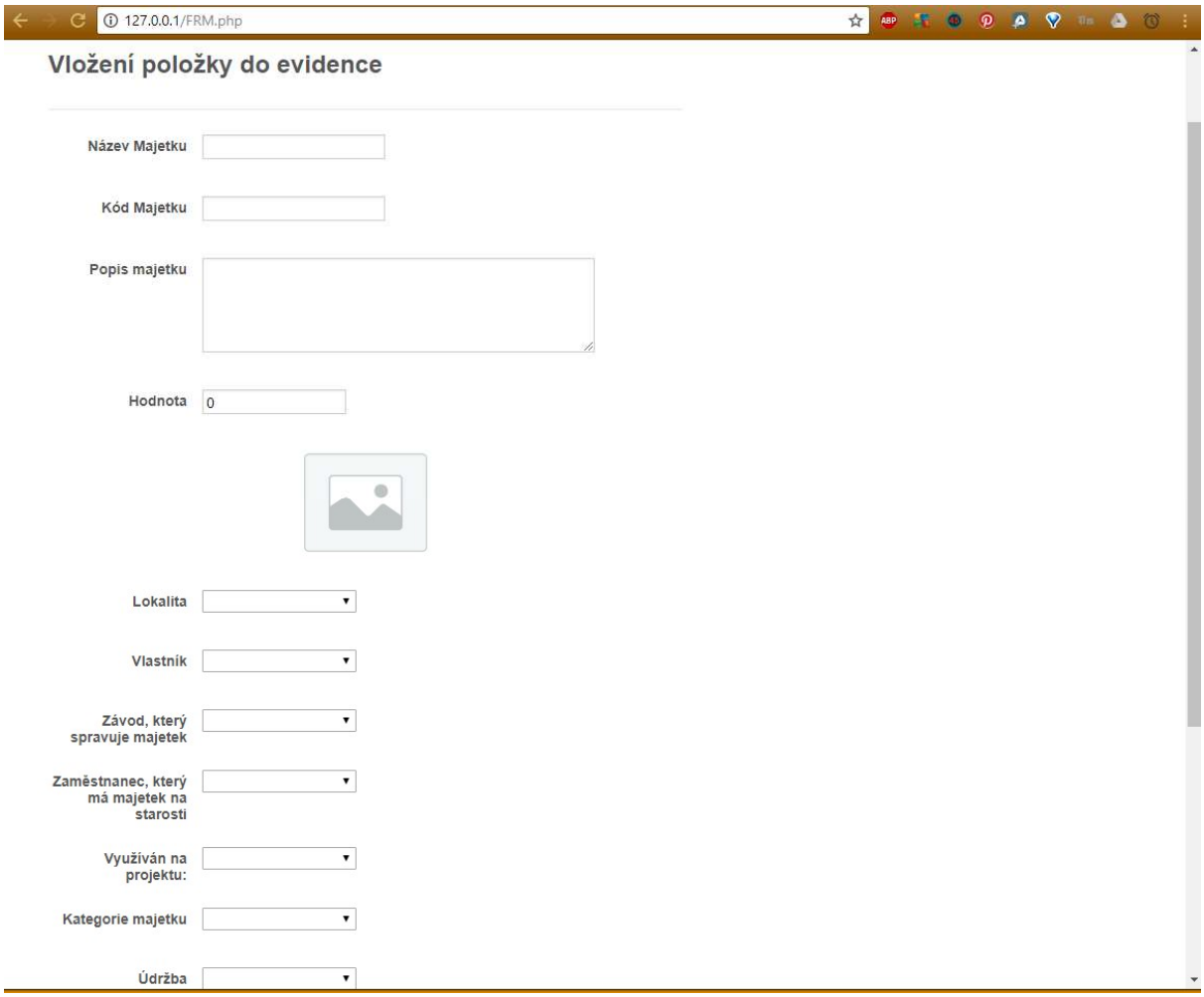
Ve webovém rozhraní „phpMyAdmin“ s českou lokalizací se řádky vloží pomocí kliknutí na odkaz „Vložit“ v horním panelu, a poté se vypíší jednotlivé informace o dané položce ve formuláři, který se pro tytu účely právě otevřel.

Po vložení položek se vypíše SQL příkaz, který byl použit pro vytvoření těchto nových položek. Ukázka tohoto SQL příkazu je na obrázku níže.



Obrázek 23 - Vložení položky do inventáře v phpMyAdmin [9]

Ne každý uživatel databáze bude mít přístup do rozhraní, jako má správce databáze. Pro uživatele se musí vytvořit formuláře, které budou požadavky databáze. Takovýto formulář se vytvoří v rozhraní HTML a pomocí příkazových řádků v souboru ve formátu PHP se poté přepíší data v databázi.



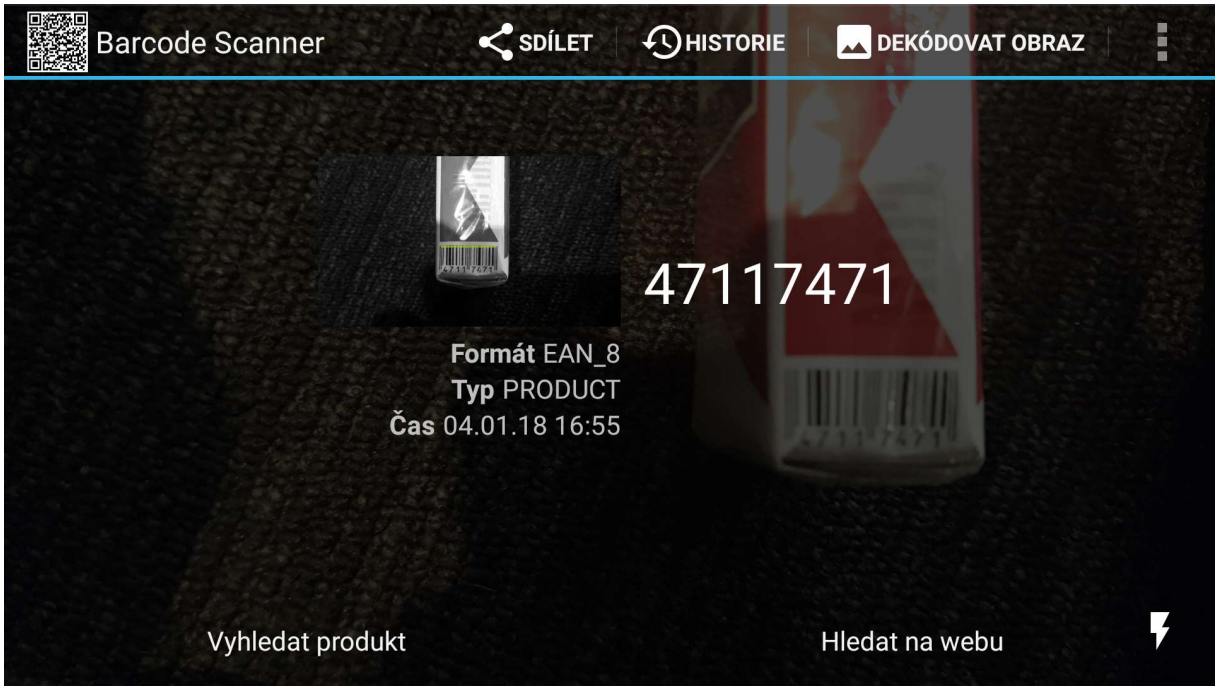
Obrázek 24 - Ukázka formuláře na vložení položky do evidence [vlastní zpracování]

Těmito formuláři, které budou dostupné ve webovém rozhraní na počítačích, ale také na mobilních telefonech, se zajistí propojení s databází tímto se taktéž bude udržovat v aktuálním provozu. Na internetu je již mnoho vzorových formulářů, jejichž skript je možno volně stáhnout a poupravit tak, aby databáze byla s PHP soubory formulářů propojena. V mém případě jsem pro tvorbu databáze zvolil službu jotform.com.

7.3 Urychlení zápisu pomocí čárových kódů

Další část je spíše praktickým doplněním k samotnému užívání databáze. Každá položka evidence by měla mít nezaměnitelný kód, primární klíč, díky kterému se můžou dohledat veškerá související data k dané položce v evidenci. Kdyby ovšem uživatel vypisoval při vyhledávání položky v databázi, při přesunu položky z jedné lokality na druhou či během jiného procesu práce s databází, chybovost při zápisu je relativně vysoká. Tato chybovost se limitně omezí použitím čárových kódů.

Při vyhledávání se může použít skener čárových kódů nebo mobilní aplikace pro vypsání hodnoty kódu. Tento kód může být vypsán pomocí skriptu v PHP do řádku ve formuláři, který může být ve formátu HTML ve webovém rozhraní. Jediným požadavkem při čtení kódu je mít kurzor na daném řádku formuláře, který má být vypsán. V dnešní době jsou dostupné freeware aplikace, které tyto čárové kódy jsou schopny dešifrovat jen díky požití té aplikace a foťáku. Jenou aplikací za všechny může být aplikace „Barcode Scanner“, která, jak je vidět na obrázku níže, okamžitě vrátí čárový kód produktu.



Obrázek 25 - Screenshot mobilní aplikace Barcode Scanner při čtení čárového kódu [10]

Pro správné užití čtení čárového kódu se musí uvažovat i nad tím, jaký formát daný čárový kód bude mít. Většina čteček je již dnes schopna rozeznat, o jaký typ čárového kódu se jedná, je ovšem vhodné uvažovat i nad formátem daného indexu majetku, primárního klíče, které bude tu danou položku charakterizovat.

Pro vnitropodnikové účely se zdá být nejvýhodnější využití dvou kódů, které podporují celou škálu abecedy latinského písma a arabských číslic s rozšířením o některé speciální symboly. Těmto kódům odpovídají:

- Code 39 – podporující 0-9, velká písmena A-Z a několik dalších symbolů. Celový počet těchto symbolů je podle názvu 39.

- Code 128 – podporující 0-9, malou abecedu a-z a velkou abecedu A-Z a další rozšiřující symboly s celkovým počtem 128.

Pro sledování jednorázových či spotřebních položek je vhodné mít čtečku EAN8, EAN13, PDF417 a QR-kódů, které jsou často využívány v obchodních řetězcích a u dodavatelů.

Takto vypadající navržený systém čárových kódů jednoznačně urychlí proces výpisu dat do formulářů evidence.

8 Závěr

Evidence majetku je důležitým prvkem nejen stavebního podniku. Její správný návrh může natolik ulehčit a zefektivnit procesy v řízení projektů za účelem snížení nákladů. Může být vhodným prostředkem pro plánování, což je klíčový proces v efektivním stavebním podnikání a realizaci projektů. Evidence majetku hlavně pracuje ve stavebním podniku s mechanizací, jež se dá považovat za investici za účelem zisku. Alokace této mechanizace k projektů, zamezení ztrát, řešení oprav a plánování přesunů mechanizace, to jsou všechno procesy, které může evidence napomoci řešit.

Cílem práce bylo vytvoření alespoň základu databáze pro evidenci majetku. Po pochopení teoretického principu navrhování databáze a postupu, při kterém se postupně vyhodnocují požadavky na databázi s pravidly tvorby, se dospělo k modelu, který je schopen evidovat majetek ve stavebním podniku. Tento model má centrální tabulku, ve které je zapsán majetek, a z této tabulky formou cizích klíčů je propojen s tabulkami, které definují vlastnosti majetku, jakými jsou ku příkladu jeho lokalita, stav, správce nebo i informace o dodavateli. Vztahy mezi tabulkami jsou řešeny formou relací tak, jak je to při vývoji podobných systémů vyžadováno.

K výběru databázového systému, ve kterém daná databáze funguje, je využito vylučovací metody a porovnání jednotlivých systémů. K výsledku ale je nutné dodat, že velkou roli hrají jednotlivé osobní priority každého správce databáze. Každý ze tří ve finální fázi porovnávaných systémů má své nesporné výhody a zároveň se již mezi sebou tolik neliší, aby zkušený programátor měl významný problém s užíváním jiného systému z těchto tří, než na který je zvyklý.

V postupném vývoji návrhu a výběru softwaru dochází k samotnému vývoji databáze tvorbou jednotlivých tabulek na základě předem navrženého modelu. Samotný proces využívá několik programů, které napomáhají k rychlejšímu a přehlednějšímu vývoji databáze. Využití těchto programů napomáhá k efektivnější práci a zároveň vychytání určitých chyb v programování. Vývoj a testování probíhalo na lokálním disku daného počítače pomocí sady vývojářských programů, které vytvoří pro databázi fiktivní lokální server.

Po vytvoření databáze je potřeba vytvořit způsoby, jak s ní bude pracovat samotný uživatel databáze. To se zajistí pomocí formulářů, které dají možnost

provádět jen povolené změny v databázi. Touto možností se zajistí životaschopnost databáze, která bude tímto způsobem mapovat reálné procesy na skladě a na stavbách. Problém, kterým může být časová náročnost ukládání a změny dat v databázi, se řeší v kapitole o urychlení procesů v databázi pomocí čárového kódu. Cílem databáze je, aby byla co možná nejvíc aktuální a přesná, ale zároveň aby uživatel netrávil příliš moc času prací nad ní.

Když bych vyhodnotili dosavadní postup při tvorbě této databáze, musím konstatovat, že se jedná o základ, na kterém se rozhodně dá stavět, ale má ještě hromadu pater před sebou. Navíc se nikdy nebude jednat o projekt, který bude někdy na 100 % dokončen, neboť bude vždy možnost tento systém rozšířit nejen nová data, ale i na příklad o analytické metody, které datům dávají smysl v kontextu reality, kterou popisují.

Jednoznačným dalším krokem ve tvorbě této databáze je samotný sběr dat, a to v kontextu reálného podniku, který tento databázový systém bude využívat. Poté se mohou aplikovat statistické a analytické metody, které sebraným datům budou dávat smysl a vysvětlovat trendy v podniku, které může manažer využít. Těmito trendy mohou být informace o využívanosti jednotlivých položek majetku, jejich přesuny, opravy a jiné informace, které povedou k lepší organizaci podniku a ušetření nákladů.

Citovaná literatura

[1] Oracle Corporation, „MySQL Workbench," 2018. [Online]. Available: www.mysql.com/products/workbench/.

[2] T. Niemi, J. Nummenmaa a P. Thanisch, Normalising OLAP cubes for controlling sparsity., Data Knowl. Eng., 2003.

[3] S. Rizzi, Business Intelligence. Encyclopedia of Database Systems, Springer Science+Business Mediea, LLC, 2009.

[4] S. Rizzi, Research in data warehouse modeling and design: dead or alive?, Arlington, Virginia: ACM, 2006.

[5] Database Answers Ltd., 2017. [Online].

[6] C. Inc., „The Top 7 Free and Open Source Database Software Solutions," [Online].

[7] solid IT gmbh, „DB-engines ranking," 2018. [Online]. Available: https://db-engines.com/.

[8] VertrigoServ WAMP Server, „Bezplatné (FreeWare) webové prostředí," 2018. [Online]. Available: https://www.vswamp.com/?lang=cz.

[9] Project PhpMyAdmin, „phpMyAdmin," 2018. [Online]. Available: https://www.phpmyadmin.net/.

[10] ZXing Team, „Barcode Scanner," 2018. [Online]. Available: https://play.google.com/store/apps/details?id=com.google.zxing.client.android&hl=cs.

[11] Construction Financial Management Association, Financial Management and Accounting for the Construction Industry, sv. Volume 1, LexisNexis, 2005.

[12] Project Management Institute, A Guide to the Project Management Body of Knowledge (PMBOK Guide), Newton square, Pa: Project Management Institute, 2004.

[13] P. Ing. Jan Tyrychtr, Disertační práce: Metodika návrhu multidimenzionální databáze v prostředí zemědělského podniku, Praha: ČZU, 2013.

[14] J. Gray, „"Evolution of Data Management"," *Computer v29 n10,* Říjen 1996.

[15] Charles Babbage Institute and others, „UNIVAC Conference Oral history," Minneapolis, 1990.

Seznam obrázků

Seznam tabulek