Databázový systém pro správu veterinární stanice

Tomáš Habrovanský

Bakalářská práce
2006

Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky

Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky
Ústav aplikované informatiky
akademický rok: 2005/2006

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení: **Tomáš HABROVANSKÝ**
Studijní program: **B 3902 Inženýrská informatika**
Studijní obor: **Informační technologie**

Téma práce: **Databázový systém pro správu veterinární stanice.**

Zásady pro vypracování:

1. Studium prostředí databázového systému MS ACCESS 2000.
2. Analýza stavu problematiky správy a zásobování veterinární stanice.
3. Návrh struktury efektivního databázového systému.
4. Vytvoření systému, který bude obsahovat:
- kartotéku všech pacientů, jejich majitelů a úkonů s pacienty,
- fakturační systém.
5. Správa a zabezpečení vytvořené databáze.

Rozsah práce:

Rozsah příloh:

Forma zpracování bakalářské práce: **tištěná/elektronická**

Seznam odborné literatury:

Halvorson Michael: Microsoft Visual Basic 6.0 Krok za krokem
Feddema Helen: Mistroství v Microsoft Access 2002
Morkes David: Microsoft Access 2002
Hernandez Michael J., Viescas John L.: Myslíme v jazyku SQL knihovna programátora
Fikáček, I., Rozehnal, I., Fikáček, M: Access 2000 – podrobný průvodce začínajícího uživatele, Grada Publishing, 1999.
Písek, S.: Databáze v Accessu, Grada Publishing, Praha, 2003.

Vedoucí bakalářské práce:	**Ing. Zdenka Prokopová, CSc.** Ústav aplikované informatiky
Datum zadání bakalářské práce:	**14. února 2006**
Termín odevzdání bakalářské práce:	**16. června 2006**

Ve Zlíně dne 14. února 2006

prof. Ing. Vladimír Vašek, CSc.
pověřený děkan

L.S.

doc. Ing. Ivan Zelinka, Ph.D.
ředitel ústavu

ABSTRAKT

Tento databázový systém je určen především pro správu privátní veterinární stanice, která ošetřuje ustájená zvířata nebo pro farmy. Systém obsahuje kartotéku všech pacientů, ve které jsou shrnuty jak celkové zdravotní údaje pacienta, tak i jednotlivé lékařské zákroky pro každé vyšetření. Jednotlivá zvířata lze zahrnout do vybrané farmy nebo stáje a určit jejich majitele.

Databázový systém je vytvořen v databázovém prostředí Microsoft Access. Pomocí propracovaného systému formulářů lze proniknout do všech obsažených dat databáze a efektivně tak zpracovávat zdravotní údaje pacienta a vytvářet výměry pro fakturaci pro každého majitele zvířat, do kterého lze zahrnout všechny provedené vyšetření jeho pacientů.

Klíčová slova:

Databáze, tabulka, formulář, SQL, Microsoft, Access, veterinární stanice, lékař, pacient

ABSTRACT

This Database system is defined for an administration of private veterinary ambulance, which cures stabling animals or for farms. The system includes a card-index of all patients, where are summaried general health data of patients and medical interventions for every investigations. A vet can implicate animals into a selected farm or a stable and determine their holders.

Database system is created by means of database interface the Microsoft Access. Uses can access to all data, which is includes in this database, by forms and they can elaborate health data of patients and create an invoice for each holder. He can include all executed investigation of his patient there, too.

Keywords:

Database, table, form, SQL, Microsoft, Access, veterinary station, vet, patient, ambulance

Chtěl bych na tomto místě především poděkovat veterinárnímu lékaři MVDr Janu Fillovi, díky kterému vznikl tento projekt a pomohl mi proniknout do problematiky veterinární praxe.

Dále bych rád poděkoval svým spolužákům Pavlovi Novákovi a Dušanu Janováčovi, kteří mi pomohli ve studiu a často mi ukázali cestu jak jít dál.

Nakonec patří mé díky manželce Lidce Habrovanské, která má se mnou bezmeznou trpělivost při mém studiu na vysoké škole a díky ní jsem došel až k tomuto okamžiku…

OBSAH

ÚVOD

V této práci vznikl databázový systém, který může využívat veterinární stanice zaměřená především na chovná zvířata ve větších stádech, jako jsou chovy koní, farmy pro dobytek, ale i pro lékaře ošetřující zvířata v zoo. Aplikační prostředí systému a návrh tabulek databáze je přizpůsoben více lékařům, kteří spíše ordinují v terénu a ošetřují vetší počet zvířat a pracují spíše pro podniky než pro drobné chovatele. Pro menší ordinace drobných domácích zvířat není program přesně přizpůsoben na míru, nicméně po menší úpravě formuláře pro vykázání výměru by se mohl program stát univerzální pro všechny veterinární praxe.

Databáze je vytvořena v databázovém systému Microsoft Access z rodiny balíčku MS Office. Volba systému byla poměrně jednoduchá, protože jak už bylo řečeno je součásti MS Office a tudíž pro uživatele operačního systému MS Windows poměrně snadno dostupná. Jeho instalace se může stát součástí dalších produktů tohoto balíčku a neklade žádné nároky na uživatele. Další výhodou je, že je spousta uživatelů operačního systému MS Windows zvyklá na filozofii vzhledu a ovládání aplikací vytvořené společnosti Microsoft, proto neklade vytvořená databáze velké nároky na zaškolení uživatele a program nemusí obsahovat rozsáhlý systém nápovědy.

MS Access je rovněž nakloněn tvůrcům aplikace. Vytváření uživatelského rozhraní je jednoduché a pomocí velkého množství profesionálně propracovaných průvodců můžeme vytvářet různé prvky aplikace. Požadavky vývojářů, které nedokáží zvládnout průvodci mohou být zpracovány v návrhovém prostředí a prvky databáze obslouženy dotazovacím jazykem, makrem nebo programovým kódem zapsaným ve Visual Basicu. Efektivní zpracovaní velkého množství dat je další předností.

Vlastní aplikace prezentuje data tabulek databáze uživateli prostřednictvím formulářů, které jsou s nimi svázány. Vytvořené sestavy pak uživateli nabízí přehledný seznam poskytnutých dat, které lze i vytisknout. Pro operace s daty jsou v mnoha případech používány dotazovací tabulky, na které navazuje programový kód VBA. Pro uživatele jsou však tyto nástroje ukryté a ten tak může přehledně nakládat pouze s prostředky, které jsou pro jeho práci nezbytné.

Cílem této práce bylo nejen vytvořit funkční a efektivní aplikaci, ale i seznámit se s některými prostředky a strukturou databázového stroje Microsoft Jet a s principy

objektové databázové hierarchie DAO. Výsledkem jsou cenné znalosti v oblasti tvorby databázových systémů, ale hlavně je vytvořena aplikace, která tvoří základ pro správu veterinární stanice. Již nyní je jisté, že aplikace splňuje základní požadavky uživatele, ale na druhou stranu si ještě vyžádá zdokonalení o další prvky uživatelského rozhraní a ve struktuře databáze.

I. TEORETICKÁ ČÁST

1 TEORIE RELAČNÍCH DATABÁZÍ

1.1 Databáze

Databází rozumíme prakticky cokoliv co obsahuje množinu dat. Databáze zahrnuje nejen vlastní data, která vyžaduje uživatel, ale i jejich vlastnosti, strukturu, omezení, relační vztahy mezi prvky databáze, integritní omezení a také aplikační rozhraní, které zprostředkovává data uživateli.

Databázi tak může tvořit několik částí. Datovým modelem můžeme chápat popis problému databáze tak, jak vychází z reálného světa. Můžeme tak definovat atributy databáze a přidělovat jim vlastnosti. Můžeme zde popsat vztahy mezi entitami nebo vazby datového systému. Přidělením fyzického rozvržení databáze a implementování tabulek nám vznikne databázové schéma. Fyzickou implementací dat se obvykle nemusíme zabývat, protože nám tuto práci obstará databázový stroj.[8]

1.2 Databázový stroj

Databázový stroj nám může pomocí programového kódu nebo jiného vývojového prostředí vytvořit fyzické objekty, pomocí kterých budeme ukládat a spravovat data. Databázové stroje vytvářejí nástroje pro fyzickou manipulaci s daty. V tomto projektu budeme hovořit především o databázovém stroji Microsoft Jet. Tento systém využívá také Microsoft Access, který vytváří zprostředkovatele mezi databázovým strojem, tvůrcem databáze a uživatelem.

Pomocí Microsoft Access můžeme vytvořit aplikaci, která obsahuje prvky formuláře nebo dialogového okna, které můžeme svázat se zdrojem dat. MS Access obsahuje datový objektový model, který propojí aplikaci s databázovým strojem. Objektovým modelem definujeme množinu objektů, které obvykle obsahují stejné vlastnosti jako data, která obsluhují.[7]

1.3 Relační databázový model

Na základě matematických principů odvozených z teorie množin a predikátové logiky vznikl model relačních databází. Otcem této myšlenky se stal E.F Codd, který v roce 1970 publikoval své výsledky v oblasti datového modelování. Dokázal, že relačním modelem

můžeme reprezentovat data, jejich strukturu i operace nad daty. Díky tomuto modelu můžeme efektivně vytvářet a spravovat databázi. Při vytváření relačních databází se musíme řídit minimálními podmínkami relačnosti. První podmínkou je, aby veškerá data byla uložena v tabulkách. Dále musí být struktura údajů a uložení dat na sobě nezávislé a uživateli není možné poskytnout přístupové cesty. A poslední základní podmínkou je, že uživatel k datům přistupuje pomocí databázového jazyka, programového kódu nebo jiného uživatelského rozhraní.[5]

1.4 Tabulka

Základem reprezentací dat v relačních databázích jsou tabulky. Tabulka je vytvořena řádky a sloupci. Řádky a sloupce tak vytvoří matici buněk. Jednotlivá buňka souvisí s právě jedním řádkem a jedním sloupcem. Buňka tak vytvoří základní entitu databáze.

Sloupce tabulky obvykle určují typ a vlastnosti společného prvku databáze a fyzickou strukturu jednotlivých záznamů v tabulce a stávají se tak atributem tabulky. Řádky tabulky vytvářejí množinu záznamů databáze. Řádek tabulky souvisí s jedním záznamem a sloupce udávají jejich jednotlivé vlastnosti.

1.5 Primární klíč

Aby bylo možné jednoznačně identifikovat jednotlivé záznamy tabulky musí být alespoň jeden atribut záznamu označen jako primární klíč. Primárním klíčem může být také kombinace několika sloupců tabulky. Hlavní podmínkou je, aby primární klíč byl v každém záznamu jedinečný. Znamená to tedy, že žádný záznam nesmí sdílet stejnou hodnotu primárního klíče s jiným záznamem. Tato hodnota zároveň nesní být typu NULL, to znamená, že primární klíč musíme vždy zaplnit konkrétní hodnotou. V případě, že ukazujeme na hodnotu primárního klíče jiné tabulky jde o cizí klíč.[8]

1.6 Relační vztahy

Pokud vznikne mezi tabulkami nějaký vztah, musíme tomuto vztahu určit kardinalitu. Obecně používáme tyto typy kardinality:

1.6.1 Vztah jedna ku jedné (1:1)

V tomto vztahu náleží jeden záznam první tabulky pouze jednomu záznamu tabulky druhé. Tento model relace není příliš často používán, protože přináší dojem rozdělení jedné tabulky na více spojených tabulek. Pokud změníme počet záznamů v jedné z tabulek, můžeme porušit integritu datového modelu. Můžou tak vzniknout sirotské záznamy v jedné z tabulek, které nemají relaci se žádným záznamem tabulky související. Vztah 1:1 se používá spíše pro rozdělení tabulky s nadměrným počtem atributů nebo pro rozdělní atributů podle společných vlastností.

1.6.2 Vztah jedna ku mnoha (1:N)

Je to běžněji používaný typ vztahu, kde jeden záznam první tabulky je možné svázat s jedním nebo více záznamy tabulky podřízené. Z toho vyplývá, že v tomto vztahu musíme určit směr relace a správně popsat jeho volitelnost. Tato podmínka opět souvisí s integritou datového modelu. Pokud smažeme záznam podřízené tabulky, může nastat situace, kde prvek primární tabulky ukazuje na neexistující data. Jako příklad můžeme uvést tabulku zaměstnanců, která kromě jména a příjmení obsahuje atribut oddělení, který poukazuje na tabulku Oddělení. Tato hodnota je reprezentována hodnotou primárního klíče podřízené tabulky. Podřízená tabulka pak obsahuje další vlastní atributy.

1.6.3 Vztah mnoho ku mnoha (N:N)

Pokud dojde k situaci, že je potřeba například zapsat jednoho zaměstnance na více oddělení a v jednom oddělení může pracovat více zaměstnanců musíme vytvořit pomocnou spojovací tabulku s relací mnoho ku mnoha. Tato tabulka bude prostředníkem mezi oběma tabulkami. Do záznamu této tabulky se zapíší primární klíče z obou tabulek a vytvoří tak cizí klíč záznamu. Tabulka s tímto vztahem je spojením dvou vztahů 1:N.[5]

1.6.4 Unární vztah

Pokud máme tabulku zaměstnanců, která obsahuje atribut jméno nadřízeného, vzniká tak relace odkazující sama na sebe. Pokud určíme směr relace a kardinalitu, kde je použit vztah 1:N, může nám vzniknout jakási organizační hierarchie. Tato hierarchie nám zobrazí strom zaměstnanců, kde na vrcholu bude stát pomyslný ředitel podniku.[8]

1.7 Normalizace databází

Pokud používáme relační databázový model, musíme dodržovat normalizační pravidla tzv. normální formy. Tyto formy nám zajistí maximální odstranění redundancí dat. Čím vyšší se použije normální forma, tím lépe a efektivněji se bude s použitými daty pracovat. Nejčastěji se setkáváme se třemi případy normálních forem.

1.7.1 První normální forma (1NF)

Tabulka s první normální formou je taková, jejíž atributy jsou dále nedělitelné. Jeden sloupec nesmí obsahovat více druhů údajů. Musí tedy obsahovat jednu skalární hodnotu. Pokud tabulka nesplňuje tuto podmínku je v tzv. nulté normální formě (NFNF) a je vhodné ji rozdělit.

Rozdělení těchto atributů nemusí být vždy jednoznačné a záleží také na tom jak s těmito prvky databáze pracuje. Typickým příkladem může být atribut jména zaměstnance, které můžeme rozdělit na jméno, příjmení, popřípadě titul. Problém nastane v okamžiku, kdy jeden zaměstnanec má dvě jména. Navrhovatele databáze pak může vést myšlenka, že vytvoří ještě jeden atribut nazvaný druhé jméno. Pokud by ovšem byl tento zaměstnanec jediný nebo druhé jméno nebylo v aplikaci použito pro třídění, vzniklo by zbytečně mnoho prázdných prvků v ostatních záznamech tabulky.

1.7.2 Druhá normální forma (2NF)

Pokud splňuje tabulka podmínku 1NF a každý atribut závisí na celém primárním klíči jedná se o tabulku s druhou normální formou. 2NF se týká tabulek, které mají více primárních klíčů. Tabulka s jedním primárním klíčem je automaticky v 2NF. Pravidlem pro vytváření tabulky v 2NF je nevyjadřovat v jedné relaci dvě různé entity. Zabráníme tím redundanci dat a také zajistíme, aby relace neodkazovala na prázdnou hodnotu.

1.7.3 Třetí normální forma (3NF)

Pokud splňuje tabulka 2NF a všechny její neklíčové atributy jsou na sobě nezávislé jedná se o tabulku v třetí normální formě. Jako příklad si můžeme uvést tabulku zaměstnanců, kde kromě atributů jména a příjmení, bude i atribut město a PSČ. Pro dodržení 3NF bychom nemuseli sloupec PSČ zadávat a tuto hodnotu bychom mohli vyhledat v tabulce pro města. Pokud bychom ale měli firmu s 20 zaměstnanci, je nesmyslné vést další tabulku s tisíci městy a směrovacími čísly. Striktní dodržení 3NF tak v tomto případě vede k nadbytečnosti dat a ztrátě výkonu databáze.[5]

2 DATABÁZOVÝ SYSTÉM MS ACCESS

Databázový systém MS Access je jeden z nejpopulárnějších databázových systémů z produkce firmy Microsoft. Tento systém nabízí širokou paletu nástrojů pro vytváření databázových aplikací, která jsou svým vzhledem co nejblíže přizpůsobena zvyklostem uživatele používající operační systém z rodiny Windows.

Návrhové prostředí tohoto systému je vytvořené tak, aby si základní aplikace mohl vytvořit i pokročilý uživatel. Nemusí se tak pokaždé spoléhat na programátora, který se specializuje na databázové systémy. Access umožňuje jednoduchý přístup k datům, jejich správě a archivaci. Formuláře, které se snadno a intuitivně vytvářejí uživateli příjemně zprostředkují data uložená v tabulkách databáze.

MS Access je součástí balíku MS Office, jehož aplikace sdílí společný programovací jazyk Visual Bacic for Applications. Bez dalších finančních nákladů a nutnosti instalací je nám tak k dispozici databázový systém, který plně podporuje export dat mezi jednotlivými sesterskými aplikacemi balíčku MS Office.

2.1 Architektury uživatelského rozhraní

Architektury uživatelského rozhraní můžeme shrnout do dvou skupin. Jestli se v aplikaci objevuje pouze jedno okno, pak hovoříme o rozhraní jednoho dokumentu (Single Document Interface SDI), nebo aplikace obsahuje základní okno a pomocí něho můžeme otevírat okna závislé na hlavním otevíracím okně. Takový dokument se nazývá rozhraní více dokumentů (Multiple Document Interface MDI). Nemůžeme dopředu říct, který z těchto dokumentů je pro uživatele výhodnější. Záleží spíše na modelu aplikace a na pracovních procesech, které má aplikace podporovat.

SDI aplikace nabízí pouze jedno okno a tak doplňující informace musí navrhovatel nabídnout prostřednictvím pomocných dialogů. Model SDI se tak spíše hodí pro aplikace s jednou logickou entitou, ale na druhou stanu může podporovat více tabulek. Můžeme mít tak jeden formulář se stejnými prvky pro vyplnění více tabulek. Po přepnutí mezi tabulkami se pouze změní titulek u názvu vyplňovaného atributu. Dokumenty tohoto typu ovšem nepodporuje MS Access, nicméně po maximalizaci hlavního okna hned po spuštění aplikace a odstraněním tlačítka pro minimalizaci je možně vytvořit jakési pseudo SDI rozhraní.

Z toho vyplývá, že MS Access je orientovaný na MDI rozhraní, kde otevřeme hlavní okno a pomocí něho otvíráme okna synovská. Každé synovské okno je tedy vhodné spojit s jednou tabulkou, jejíž záznamy budou zdrojem dat pro otevřený formulář. Model MDI pak nabízí několik variant:

V klasickém pojetí modelu MDI se otevře hlavní okno a v něm se potom otevírají podokna. Takový model podporují některé produkty balíčku MS Office jako Word, Excel ale i Access. Problémem tohoto rozhraní může být nekonzistence kontejnerového modelu. Rodičovské okno obsahuje synovská okna, která jsou v něm otevřena, ale samotná aplikace nemusí obsahovat objekty, které tato okna prezentují. Nicméně pro aplikace, kde je potřeba otevření několika oken současně jsou stále dobrým řešením.

Rozhraní, která po otevření zobrazí jakýsi centrální přepínací panel obsahují na tomto panelu sadu přepínacích tlačítek. Pomocí těchto tlačítek se uživatel dostane k formuláři svázaným s určitou tabulkou. MS Access navíc prostřednictvím průvodce nabízí automatické vygenerování tohoto panelu včetně programového kódu obsluhující jednotlivá tlačítka. Toto rozhraní může být vhodné pro aplikace, kde je nutné podporovat několik různých pracovních procesů. Tlačítka přepínacího panelu tak slouží ke spouštění jednotlivých procesů.[8]

2.2 Aplikační prostředí MS Access

Uživatelské rozhraní MS Access nabízí základní databázové okno, které obsahuje několik záložek. Jednotlivé záložky obsahují databázové objekty příslušných typů jako například tabulky, formuláře nebo dotazy. Vytvořený ucelený systém objektů pak vytváří aplikační uživatelské rozhraní.

Pro větší zabezpečení dat a zachování datové integrity není vhodné ponechat tabulky volně přístupné uživateli, ale zprostředkovat pomocí dostupných objektů. Ryzí uživatel používá ke své činnosti s daty především formuláře, které mu nabízí přehlednou a bezpečnou práci s daty. Formulář je téměř vždy svázán s některou z tabulek databáze. Ve formuláři je obvykle pomocí prvků formuláře zobrazen pouze jeden aktuální záznam. Mezi záznamy pak můžeme přepínat pomocí navigačního panelu. Některé formuláře lze zobrazit v podobě datového listu, které uživateli poskytnou širší přehled několika záznamů najednou, podobně jako je tomu v tabulkách.

Tisk nebo přehledné prohlížení dat umožňují sestavy. Tyto sestavy jsou obvykle sestavěny pomocí podpůrných objektů jako jsou dotazy, moduly nebo makra. Aplikace sama řídí běh souvisejících úloh a uživatel se nemusí starat s organizací databáze.[1], [2]

2.3 Návrh vytváření aplikace

Jednodušší části aplikace lze vytvořit pomocí průvodce. Průvodců Access nabízí celou řadu v podobě profesionálně propracovaných produktů. Microsoft však nabízí prostřednictvím průvodců svou filozofii návrhu formulářů, v některých případech jsou příliš universální, a tak jsou některé možnosti vzhledu formuláře pro návrháře nedostupné. Pokud ovšem chce programátor opustit nabízenou koncepci průvodců, nezbývá mu než aplikaci vytvořit v návrhovém zobrazení.

Jako první krok je nutné vytvořit tabulky v návrhovém zobrazení, které musí být předem promyšlené, včetně relací se souvisejícími tabulkami. Předem promyšlená struktura databáze je základnám kamenem pro vytvoření funkční aplikace. Pozdější úpravy struktury databáze jsou sice možné, ale za cenu zdlouhavých procedur, které přinášejí potíže v podobě chyb. Do tabulek se vyplní jednotlivé atributy, stanoví se jejich datové typy a nastaví se jejich vlastnosti. Vytvoří se relace mezi souvisejícími tabulkami.

K jednotlivým tabulkám se vytvoří formuláře a propojí se s tabulkovými daty. V návrhovém zobrazení pro tvorbu formulářů je vhodné upravit vzhled a funkci formuláře podle požadovaných grafických vlastností zadaných uživatelem nebo návrhářem aplikace. Nastaví se vlastnosti jednotlivých prvků a připojí se na ně událostní procedury nebo makra. Do formulářů je možné vložit další prvky jako grafické objekty, ovládací tlačítka, podformuláře apod.

Vytvoří se událostní procedury a makra související s prvky formuláře. Kód VBA se ukládá do modulů, které jsou součástí formuláře. Pokud je modul uvozen klíčovým slovem Sub nevrací žádnou hodnotu. Pokud je ovšem uvozen klíčovým slovem Function, pak modul vrací hodnotu, kterou lze dále využívat v programovém kódu.[3]

2.4 Programové jazyky

I když je MS Access navržen tak, aby se uživatel z větší části obešel bez programování, poskytuje dva programové jazyky, pomocí kterých můžeme profesionálně a efektivně přistupovat k jednotlivým prvkům databáze.

2.4.1 SQL

Programový jazyk SQL je v aplikacích MS Accessu nejvíce využíván v dotazových sestavách. I když Access nabízí kvalitní a elegantní návrhové rozhraní pro vytváření dotazů, je znalost dotazového jazyka SQL výhodou při sestavování definičních, předávacích nebo sjednocovacích dotazů, nebo pro programátory, kteří tento jazyk používají i v jiných databázových systémech a mají zkušenosti s tímto kódem. V návrhovém zobrazení pro vytváření dotazů můžeme vytvořit dotaz, který si pak převedeme do zobrazení SQL a ten pak můžeme také implementovat do vlastního kódu VBA.

Každý příkaz SQL začíná jedním ze sedmi příkazů, které nám udávají typ dotazu.

Tab. 1. Tabulka základních SQL příkazů a jejich význam

Příkaz	**Význam**
ALTER	Upraví atributy tabulky
CREATE	Vytvoří tabulku
DELETE	Vymaže záznamy tabulky
DROP	Vymaže tabulku z databáze
INSERT	Vloží záznamy
SELECT	Provede zobrazení vybraných atributů tabulky
UPDATE	Aktualizuje záznamy tabulky

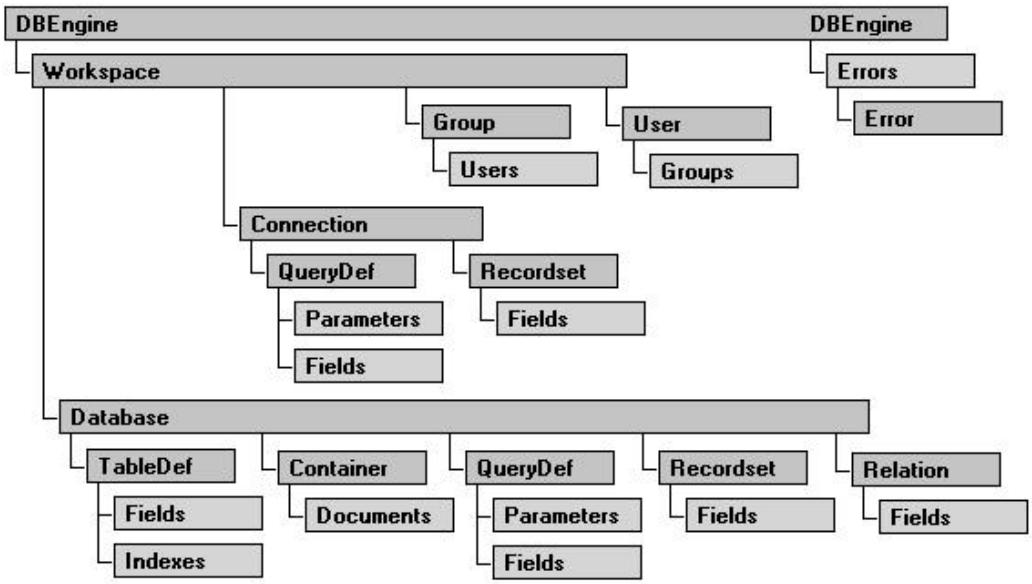
Za ním následuje klíčové slovo, FROM, WHERE, GROUP BY nebo HAVING, které specifikuje způsob provedeného dotazu. Můžeme zde použít i klíčová slova jako LEFT, RIGHT, INNER, JOIN, které slouží pro propojení jednotlivých tabulek, které mezi sebou vytvářejí relaci. Jazyk SQL také nabízí agregační výběrové dotazy, které odkazují na

relačně spojené tabulky obohacené agregačními funkcemi. Mimochodem, téměř všechny agregační funkce používané jazykem SQL obsahuje i kód VBA.[10]

2.4.2 Visual Basic for Applications (VBA)

VBA je objektově orientovaný jazyk s implementovanými objektovými typy využívající ve svých modulech typické prvky strukturovaného programování jako jsou cykly FOR, DO, WHILE, větvení programu IF – THEN, SELECT CASE stejně jako je známe z Pascalu nebo Jazyka C. Tyto objekty obsahují vlastnosti, metody a události, na které mohou objekty reagovat. Objektové typy tvoří kolekce. Jednotlivé kolekce vytvářejí hierarchii objektů databázového stroje.

Na vrcholu hierarchie VBA využívající databázového stroje je objekt DBEngine (DBE), z kterého se odvozují objekty Workspace a Error. Z Workspace se dále odvozují hierarchie databáze a hierarchie řídících objektů.



Obr. 1. Hierarchie objektů databázového stroje

Objekty DAO reprezentují strukturu databáze. Tyto objekty můžeme použít pro vytvoření a změnu tabulek a dotazů, pro zabezpečení databáze nebo pro přístup k datům v externích zdrojích dat. Můžeme je také použít pro manipulaci s daty uloženými v databázi z programového kódu.[7]

V praxi velmi často využívaným objektem je objekt DoCmd. Objekt je vybaven metodami, které slouží k provádění úloh jako například zavření okna, otevření formuláře nebo nastavení hodnot ovládacího prvku. Metody objektu DoCmd jsou také častými metodami vygenerovanými průvodci aplikace a většina z nich lze také nahradit makrem.

Jako každý vyspělý programovací jazyk obsahuje kód VBA funkce a procedury. Událostní procedury prvků formulářů vygeneruje průvodce a programátor pouze doplní kód obsluhující proceduru. Kód VBA neběží řádek po řádku, ale jeho procedury se spouštějí jako odezvy na události vzniklé během aplikace. Každý objekt má svou vlastní sadu událostí, na které dokáže reagovat.

Kromě veřejných procedur vkládáme do standardních modulů funkce. Mohou přijímat parametry a vracet funkční hodnotu. Jazyk disponuje například funkcí MsgBox, pomocí které můžeme zobrazit dialogové okno. Toto okno nabízí širokou paletu uplatnění ve formě zpráv, výzev, chybových hlášení nebo výběru hodnot.[4]

2.5 Zabezpečení databáze

2.5.1 Nastavení hesla pro otevření databáze

Pokud s databází pracuje pouze jeden uživatel, je vhodné použít jednoduchou úroveň zabezpečení databáze heslem. Pomocí dialogového okna pro nastavení hesla databáze nastavíme nové heslo. Tento dialog lze spustit pouze ve výhradním režimu, který je nutné zvolit při otevírání databáze.

Tato metoda je jednoduchá a bezpečná. Ochrana se vztahuje pouze na otevření databáze. Pokud se databáze otevře, všechny její objekty jsou uživateli přístupné.

Při volbě hesla je hodné používat následující pravidla. :

1. Příliš krátké heslo lze dříve rozkódovat. Dlouhé heslo je složité na zapamatování a při aplikaci vniká častější chyba překlepu. Doporučuje se délka 5 až 10 znaků
2. Je vhodné požívat pouze malá písmena. Klávesnice je většinou přepnuta na malá písmena velkými písmeny bychom si přidělali starosti
3. Používat pokud možno co nejmenší počet hesel. Zapisování jiného hesla pro každou aplikaci do zápisníku také může vést k odhalení hesla

4. Heslo není vhodné nikomu sdělovat, ani nejlepším přátelům. V případném prolomení hesla bychom se také mohli zlobit na nevinného
5. Jména přítelkyň, manželek, dětí nemají velký ochranný účinek

2.5.2 Uživatelská úroveň zabezpečení

V okamžiku spuštění databázového stroje se vytvoří pracovní prostor Workspaces. Po otevření kterékoliv databáze se zřídí objekt DBEngine kolekci Workspaces a výchozí objekt Workspace. V tomto okamžiku systém zjistí, zda-li pracuje ve chráněném režimu a pokud ano, vyžaduje přístupové jméno a heslo. Tím se vytvoří objekty User a Group uvnitř standardního pracovního prostoru. Až potom se vytvoří objekt Database a na základě objektů User a Group systém zjistí přístupová práva k jednotlivým objektům aplikace.

Touto obsáhlou metodou zabezpečení je nastavení různých uživatelských úrovní. Podobně se tento způsob používá při zabezpečení sítí. V systému jsou uživatelé přihlášeni do určitých skupin. Každá skupina má přidělena práva pro obsluhu jednotlivých objektů databáze. Práva můžeme přidělit i jednotlivým uživatelům. Tento způsob zajišťuje databázi z různých směrů zneužití. Pokud skupině, která vytváří aplikaci, přidělíme práva na přístup ke všem objektům a uživatelům přidělíme práva pouze k přístupu k datům výrazně ochráníme autorská práva tvůrce aplikace. Tím se také zabrání neúmyslnému poškození aplikace změnou kódu nebo vlastností objektů. Pokud ještě rozdělíme uživatele do dalších skupin, můžeme uchránit citlivá data před zneužitím.

2.5.3 Uložení databáze jako soubor MDE

Je to velmi rychlý a účinný způsob zabezpečení databáze z pohledu nechtěného poškození aplikace uživatelem. Převedením databáze do formátu MDE spustíme proces, který zkompiluje všechny moduly, odstraní z aplikace veškerý programový kód a databázi zkomprimuje. Kód programu tedy bude nadále pracovat, ale nebude viditelný z pohledu uživatele aplikace a nemůže jej tedy ani měnit. Uživatel nebude mít možnost ani vytvářet a měnit vlastnosti jednotlivých databázových objektů. Převedením do souboru MDE zabezpečíme formuláře, sestavy a programový kód aniž by bylo nutné přihlašovat uživatele a nemusí se tak vytvářet složitý uživatelský systém práv zabezpečení.[4]

II. PRAKTICKÁ ČÁST

3 ANALÝZA STAVU PROBLEMATIKY VETERINÁRNÍ STANICE

Analýza veterinární stanice zahrnuje popis veterinární stanice z pohledu informačního technologa. Popisuje veterinární stanici jako myšlenkový datový model pro budoucí zpracování návrhu struktury databáze a návrhu uživatelského rozhraní.

3.1 Veterinární ordinace

Základním prvkem veterinární stanice je ordinace. Pracují zde ošetřující lékaři, obsahuje nástroje pro lékařské zákroky, spotřební lékařské prostředky jako např. injekční stříkačky, obvaziva, skladuje léčiva, vede kartotéku pacientů. Dále může mít ordinace účetní jednotku, na kterou navazuje název organizace, adresa, IČ, DIČ.

3.2 Lékaři

Ve veterinární stanici pracuje jeden nebo několik ošetřujících lékařů. V případě, že je ve veterinární stanici více lékařů, mohou nastat z organizačního i účetního hlediska tyto varianty:

1. Jedna veterinární stanice obsahuje několik lékařů a sdílí společné účetnictví, platební výměry vykazují pod stejnou hlavičkou, používají společné vybavení a kartotéku pacientů.
2. Každý lékař pracuje pod vlastní účetní jednotkou, vykazuje vlastní platební výměry, ale s ostatními lékaři sdílí společnou ordinaci, vybavení ordinace a kartotéku pacientů.
3. Každý lékař pracuje pod vlastní účetní jednotkou, vykazuje vlastní platební výměry, má vlastní kartotéku, používá vlastní lékařské vybavení, převážně specializované. S ostatními lékaři sdílí jen ordinaci, ve které se obvykle s ostatními lékaři střídá, požívá jen část obecného vybavení veterinární ordinace a kartotéku pacientů.

Tato práce je zaměřena pro případ 1, kde je jedna stanice pod jednou účetní jednotkou.

3.3 Kartotéka pacientů

Kartotéka pacientů obsahuje souhrn všech pacientů a jejich zdravotní dokumentaci. Skládá se ze dvou částí.

3.3.1 Obecné zdravotní a osobní údaje

V této části zdravotní dokumentace pacienta jsou uvedeny osobní údaje jako jméno pacienta, druh zvířete, datum narození, pohlaví, jméno majitele, farmu nebo místo pobytu. Jsou zde dlouhodobé zdravotní údaje, anamnéza, chronické a dědičné choroby, trvalé zdravotní postižení apod. Celá tato část tvoří záhlaví zdravotní dokumentace a některé hlavní části jako jméno, druh zvířete, jméno majitele nebo farmy, by měly být zobrazovány v souhrnných seznamech pacientů.

3.3.2 Jednotlivá zdravotní vyšetření

V další části dokumentace jsou uvedena jednotlivá vyšetření pacienta. Tato vyšetření obsahují zejména: datum, důvod nebo příznaky, pro které byl pacient ošetřen, popis provedeného vyšetření, diagnózu, seznam provedených bodovaných zdravotních úkonů a seznam podaných léčiv. Datum vyšetření a popis provedeného vyšetření by měl být obsažen v souhrnném seznamu vyšetření pacienta.

3.4 Odborné zdravotní úkony

Ve veterinární praxi se stejně jako ve zdravotnictví vykazují zdravotní úkony. Tyto úkony jsou bodovány. Body každého úkonu jsou násobeny koeficientem pro všechny úkony a výsledný součin je pak účtován majiteli zvířete.

Všechny úkony jsou označeny číselným kódem. Kód je pro každý typ úkonu jedinečný, ale mnoho úkonů ještě blíže specifikují na jaké zvíře je aplikován. Například vyšetření pro koně a pro krávu mají stejný kód provedeného úkonu, ale kro každé zvíře je úkon jinak bodován.

Úkony jsou dále rozděleny do skupin. Skupiny vytvářejí souhrn úkonů podle charakteru vyšetření. Například diagnostika je v jedné skupině a chirurgické zákroky jsou zařazeny do druhé. Všechny skupiny úkonů jsou ještě rozděleny do tří oddílů: Základní vyšetření, veterinární úkony a laboratorní úkony.

3.5 Léčiva

Používaná léčiva se stejně jako úkony dělí do skupin. Tyto skupiny charakterizují o jaký druh léčiva se jedná. Například antibiotika vytvářejí jednu skupinu a vitamíny druhou. Léčiva jsou dále opatřeny číselným kódem. Tento kód je jedinečný pro každé léčivo.

3.6 Majitelé

Každé ošetřované zvíře náleží majiteli. Vztah majitele a pacienta tak úzce souvisí s platebním výměrem. Majitelem může být fyzická nebo právnická osoba. Z tohoto důvodu je vhodné pro fakturaci znát buď jméno a příjmení majitele nebo název firmy včetně identifikačních čísel a adresy.

3.7 Farmy

Zvířata jsou často shrnuta do farem nebo stájí. Veterinární stanice by měla pro lepší orientaci mít pacienty do jednotlivých stájí rozděleny. Tato skutečnost se pak může projevit jednak ve skupinovém vyšetření celého stáda, ale i v platebním výměru, kde se účtované cestovné bude vztahovat ke všem pacientům farmy. Zvláštní případ však může nastat tam, kde na jedné farmě je více majitelů. V tomto případě je vhodné jednu farmu rozdělit a dílčí farmy označit jiným identifikátorem.

3.8 Platební výměry

Platební výměry zahrnují provedené zdravotní úkony a podaná léčiva, vztahující se k pacientům, kteří mají společného majitele. Výměr je tedy adresován právě jednomu majiteli, kterému jsou počítány položky vztahující se k jeho zvířatům. Výměr čerpá platební podklady ze zdravotní dokumentace pacientů a nevykázané zákroky nabízí k fakturaci. Do výměru jsou zahrnuty náklady na cestu k pacientům a další poddodávky, které nejsou uvedeny ve zdravotní dokumentaci.

Jako příklad provedení výměru můžeme uvést následující případ: Lékař vyjel na farmu, kde ošetřil dva pacienty, u každého provedl jeden bodovaný zdravotní úkon a každému podal jednu sadu léčiv. Navíc majiteli poskytl přípravek pro každodenní aplikaci léčiva. Výměr tedy bude obsahovat: jméno majitele, dva zdravotní úkony, dvě sady podaných léčiv, cestovné a poddodávky budou obsahovat přípravek na aplikaci léčiv.

3.9 Kniha jízd a účetnictví

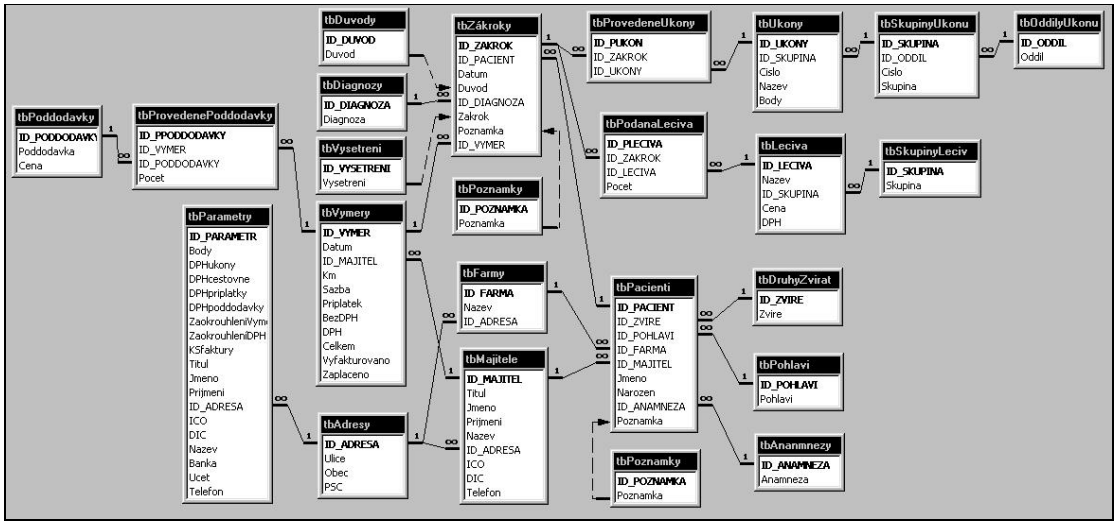
Vzhledem k tomu, že se lékaři často pohybují v terénu, měla by mít veterinární stanice knihu jízd. Kniha jízd může být pro každé vozidlo zvlášť, ale zároveň je vhodné mít souhrnný přehled o všech vozidlech, jehož obsah by se pak projevil v účetnictví stanice, nebo může souviset s platebním výměrem, kde se jednotlivá položka knihy jízd zahrne do výměru a majiteli se vykáže cestovné.

Účetnictví je celkovou finanční správou stanice. Zde se shrnují výdaje ve formě nákupu léčiv, lékařských pomůcek, lékařského vybavení stanice, režijní náklady na provoz stanice apod., dále pak příjmy ve formě vyfakturovaných výměrů a prodejů léčiv. Účetnictví zapouzdřuje dodací listy, příjmové a výdajové doklady, faktury, platby zaměstnanců, knihy jízd a daňovou správu.

Vzhledem k náročnosti účetnictví není tento modul ani kniha jízd součástí této práce. Veterinární stanice obvykle používají účetnické programy dodávané specializovanými dodavateli, nebo jim účetnictví zpracovává účetní firma. Nicméně do tohoto projektu může být přidán modul pro export dat z tabulky výměrů do programu účetnictví, který již veterinární stanice používá.

4 STRUKTURA DATABÁZE

V této kapitole je popsána struktura databáze jednak z hlediska datového modelu, kde je myšlenkový popis problému struktury databáze, tak i databázové schéma, kde je datový model popsán vůči databázi, jeho fyzická implementace do schématu dat a také aplikační pravidla.



Obr. 2. Celkové schéma tabulek databáze a jejich relace

4.1 Kartotéka pacientů

4.1.1 Tabulka pacientů

Základním kamenem pro vytvoření kartotéky pacientů je tabulka tbPacienti. Táto tabulka obsahuje základní údaje o pacientovi a je vyjádřena následující atributy:

ID_PACIENT je jednoznačný identifikátor položky tabulky, vytváří primární klíč tabulky. Typem tohoto atributu je automatické číslo, které vygeneruje databázový stroj Microsoft Jet. Tento atribut využívá pro svůj obsah tabulka tbZakroky

ID_ZVIRE uvádí o jaký druh zvířete se jedná. Druh zvířete je nutné vybrat z tabulky tbDruhyZvirat. Tato tabulka obsahuje souhrn všech druhů zvířat, se kterými se může uživatel setkat. Ke každému zvířeti je přiděleno kódové označení, které se skládá z čísel i písmen, proto je jeho datový typ text. Tento číselný kód je pro každé zvíře jedinečný, proto se také stává primárním klíčem.

ID_POHLAVI je atribut, který vyznačuje v podstatě dvě hodnoty: samec a samice. Proto by bylo z hlediska úspory dat vhodné určit jako datový typ Ano/ne (True/False). V aplikaci dat by potom Ano reprezentovalo samce a Ne samici nebo naopak. V tomto případě by musel být program ošetřen o nutnost zadávání této položky. Z důvodu jednodušší aplikace dat k uživateli je vytvořena tabulka tbPohlaví, ze které si uživatel vybere potřebná data nebo ponechá pole prázdné. Tato tabulka obsahuje pouze dvě položky a je v aplikačním prostředí uživateli nedostupná.

Protože pacient náleží farmě, u které evidujeme nejen její název, ale i ostatní údaje je položka ID_FARMA v relaci na tabulku tbFarmy stejně jako položka ID_MAJITELE, která souvisí s tabulkou tbMajitelé.

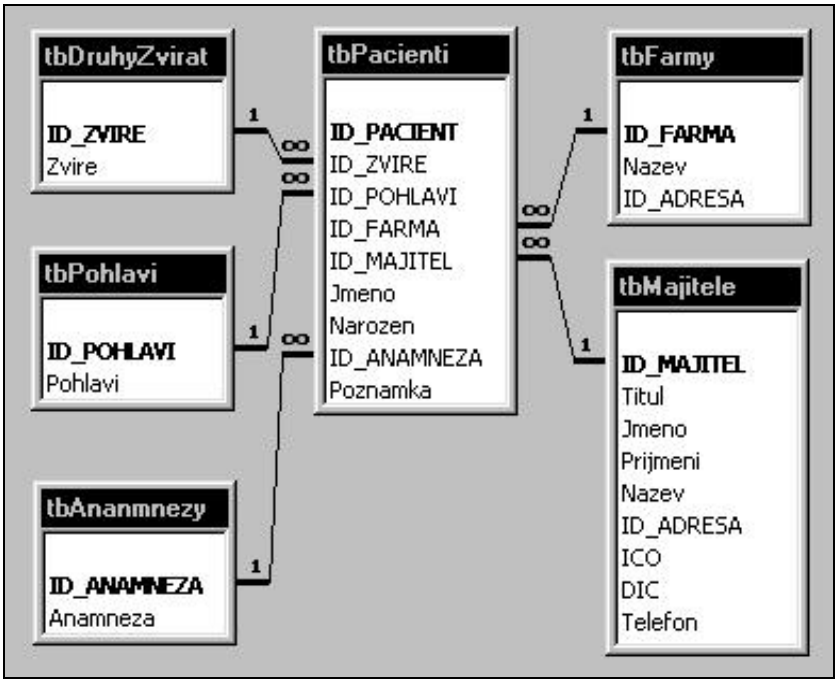
Další tabulkou, která úzce souvisí s kartotékou pacientů je tbZakroky. Obsahuje všechna vyšetření všech pacientů.

Atribut Jmeno vyjadřuje jméno pacienta.Toto pole je tedy datovým typem text o délce 50 znaků.

Atribut Narozen uvádí datum narození zvířete. Datovým typem je datum ve: tvaru den.číslo měsíce.rok.

Anamnézu zvířete je nutné zvolit pomocí tabulky tbAnamnezy přes atribut ID_ANAMNEZA. V této tabulce je zadáván název a číselný kód anamnézy, který se generuje automaticky.

Pomocí atributu Poznamka může uživatel zapsat jakoukoliv textovou poznámku. Zajímavé je, že není mezi tabulkou tbPoznamky a tbPacienti vytvořena relace. Poznámku je totiž možné vybrat pomocí formuláře z tabulky tbPoznamky nebo přímo zapsat do této položky poznámku, která není v tabulce poznámek uvedena. Tabulka poznámek tak vytváří jakýsi seznam automatických vět.



Obr. 3. Relace mezi tabulkou tbPacient a souvisejícími tabulkami

4.1.2 Tabulka lékařských zákroků

Tabulka tbZakroky obsahuje všechna vyšetření všech pacientů. Z této tabulky lze tedy vyčíst, jaká vyšetření byla pro jednotlivé pacienty provedena například pomocí dotazu qryVyberZakroku. Tabulka obsahuje následující atributy:

ID_ZAKROK je jednoznačný identifikátor položky tabulky, vytváří primární klíč tabulky. Typem tohoto atributu je automatické číslo. Atribut je dále využíván tabulkami tbPodanaLeciva a tbProvedeneUkony, které určují jaká léčiva a úkony jsou k lékařskému vyšetření přidělena. Tyto tabulky jsou typickým příkladem tabulky typu N:N, kde primárním klíčem tabulky může být kombinace dvou atributů. Nicméně pro jednodušší aplikační řešení je do těchto tabulek přidělen identifikační atribut ID, který je automaticky generován.

ID_PACIENT určuje pacienta, kterému je lékařský zákrok přiřazen. Vytváří relaci s tabulkou pacientů. Hodnota tohoto atributu je v aplikaci automaticky dána výběrem pacienta ve formuláři frmPacienti a následném otevření formuláře frmZakroky.

Datum určuje datum vyšetření a jeho výchozí hodnota je nastavena automaticky na dnešní datum. Nicméně uživatel má právo toto datum podle své vůle měnit.

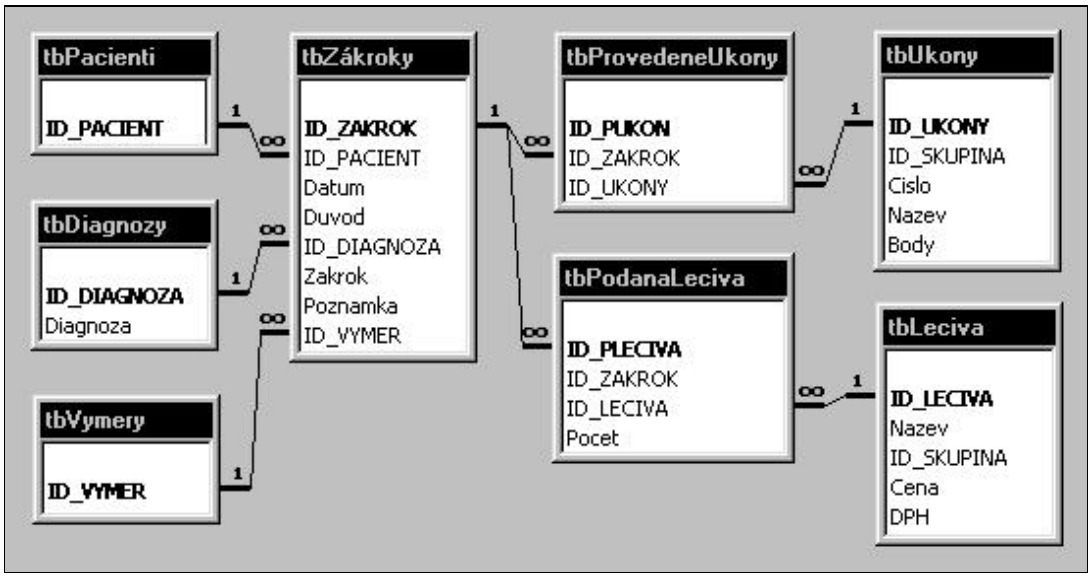
Duvod vyjadřuje důvod návštěvy pacienta nebo jeho prezentaci jeho zdravotních potíží. S tímto atributem souvisí tabulka tbDuvody.

Atribut Zakrok popisuje způsob provedení lékařského zákroku, jeho průběh nebo specifické projevy. S tímto atributem souvisí tabulka tbZakroky.

Význam tributu Poznamky je podobný jako v tabulce tbPacienti. Stejně jako atributy Duvod a Zakrok nemají relaci s příslušnými tabulkami. Související tabulky pouze nabízejí jakýsi automatický text pro rychlý výběr nebo zápis ve formuláři.

ID_DIAGNOZA je nutné vybrat z tabulky tbDiagnozy. Tato tabulka obsahuje pouze identifikátor generovaný automaticky a název diagnózy.

ID_VYMER označuje, kterému výměru je vyšetření vykázáno. Je tak v relaci s tabulkou tbVymery. Jako výchozí hodnota této položky je nastavená 0. Tato položka je pak ve formuláři pro výkaz výměrů frmVymery přepisována číslem výměru nebo naopak nulou, je-li z výměru vyřazena.



Obr. 4. Relace tabulky tbZakroky se souvisejícími tabulkami

4.2 Číselníky

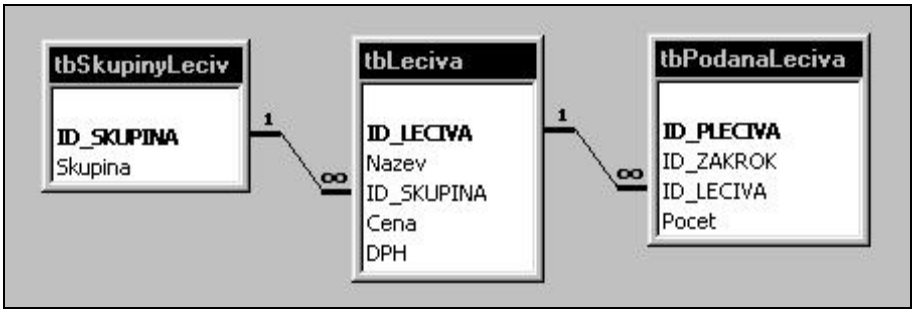
4.2.1 Tabulka léčiv

Tabulka tbLeciva obsahuje všechna léčiva, které lékař aplikuje pacientům. Atributy tabulky jsou:

ID_LECIVA určuje jedinečný kód léčiva, který může být složen z číslic i z písmen. Z tohoto důvodu je datovým typem text o délce 6 znaků. Tento atribut je tak primárním klíčem tabulky a souvisí s tabulkou tbPodanaLeciva.

ID_SKUPINA určuje, do které skupiny léčiv lék náleží a je v relaci s tabulkou tbSkupinyLeciv. Tato tabulka obsahuje atributy ID_SKUPINA, který je relačně spojen s tabulkou léčiv a název skupiny.

Atribut Nazev uvádí obchodní název léku. Cena jednoho léku je uváděna v korunách a DPH v procentech.



Obr. 5. Struktura tabulek s léčivy

4.2.2 Tabulka úkonů

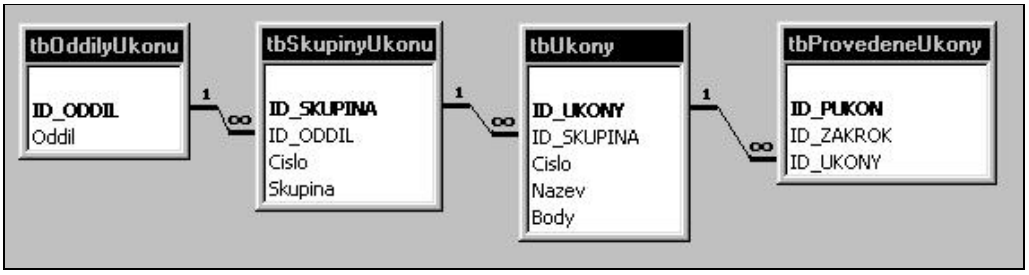
Tato tabulka obsahuje všechny bodované úkony, které lékař vykonává. Obsahuje atributy Nazev, datový typ text a Body, datový typ desetinné číslo. Další atributy vytvářejí datovou strukturu třídění úkonů:

ID_UKON je atribut, který jednoznačně identifikuje položku v tabulce. Je generována automaticky a vytváří primární klíč tabulky. Tuto položku využívá tabulka tbProvedeneUkony.

Cislo určuje číselný kód úkonu. Datovým typem je text, protože číselný kód může obsahovat před platnou číslicí nulu. Každý úkon je sice označen číselným kódem, ale některé úkony ještě blíže specifikují, na kterém zvířeti je proveden. Stejné vyšetření pro krávu nebo pro koně je sice stejně označeno kódem, ale jinak bodováno. Z tohoto důvodu se atribut Cislo nestal primárním klíčem tabulky.

ID_SKUPINA souvisí s tabulkou tbSkupinyUkonu, kde je také primárním klíčem. Tato tabulka obsahuje atributy Skupina, kde je uveden název skupiny a Cislo, který určuje číselný kód skupiny. Skupiny jsou dále rozděleny do oddílů. S tabulkou tbOddílyUkonu

spojuje tabulku tbSkupinyUkonu atribut ID_ODDIL, který je také primárním klíčem tabulky. V každém oddílu může být skupina se stejným číslem, z tohoto důvodu není atribut Cislo v tabulce tbSkupinyUkonu primárním klíčem tabulky.



Obr. 6. Struktura tabulek s úkony

4.2.3 Tabulka majitelů

Majitelé pacientů jsou obsaženi v tabulce tbMajitele. Tato tabulka by měla obsahovat údaje jak pro fyzickou, tak i pro právnickou osobu. Proto má atributy Titul, Jmeno, Prijmeni, Nazev (název firmy), IČ, DIČ. Dále pak kontaktní údaje jako:

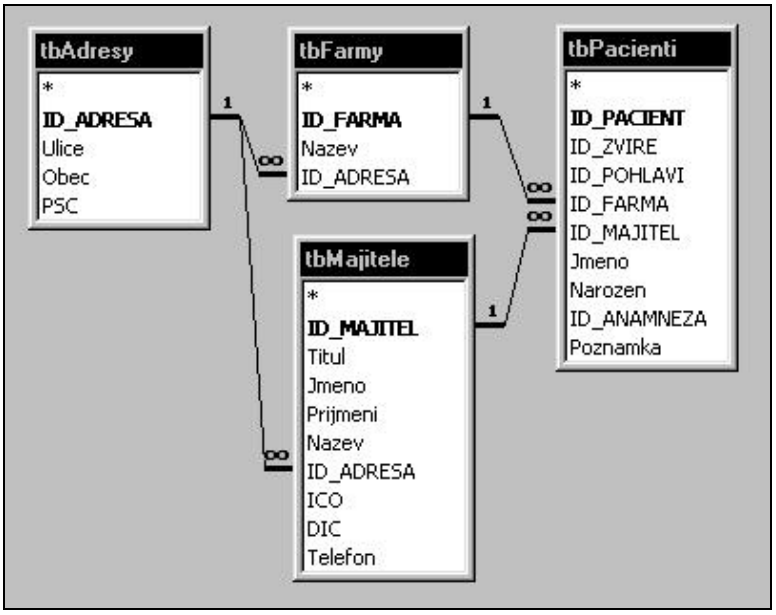
Telefon, v tomto poli lze uvést i více telefonů, proto je datovým typem text, aby bylo možné do tohoto pole napsat čárku.

ID_ADRESA je nutné vybrat z tabulky tbAdresy, která obsahuje další údaje: Ulice, Obec, PSČ.

Atribut ID_MAJITELE je primárním klíčem tabulky, generuje se automaticky a vytváří relaci s tabulkou tbPacienti a také tbVyměry.

4.2.4 Tabulka farem

Farmy jsou obsaženy v tabulce tbFarmy. Jejími atributy jsou Nazev, ID_ADRESA, která souvisí s tabulkou tbAdresy a ID_FARMY, který je primárním klíčem tabulky a generuje se automaticky.



Obr. 7. Relace související s tabulkou tbMajitele a tbFarmy

4.3 Výměry

Výměry pro fakturaci jsou obsaženy v tabulce tbVymery s těmito atributy:

ID_VYMER je jednoznačným identifikátorem položky tabulky a zároveň primárním klíčem. Tato hodnota je zadávána uživatelem a zároveň vyjadřuje číslo výměru, které lze používat také i jako číslo faktury nebo variabilní symbol. Vzhledem k tomu, že datovým typem je textová hodnota, záleží jen na uživateli, jaký systém číslování používá. Aplikační prostředí nabízí číslo o jednu vyšší než je nejvyšší hodnota pole pokud jde o ryze číselné hodnoty.

Položku ID_MAJITEL je nutné zadat z tabulky majitelů. Po zadání majitele v aplikačním prostředí nabídne formulář všechny nevykázané záznamy provedených úkonů a podaných léčiv související s uvedeným majitelem. Záleží pak na uživateli, jestli nabídnuté záznamy použije pro vykázání výměru.

Atribut Datum vyjadřuje nejen datum vykázání výměru, ale zároveň i datum splatnosti a datum uskutečnění zdanitelného plnění na faktuře související s výměrem.

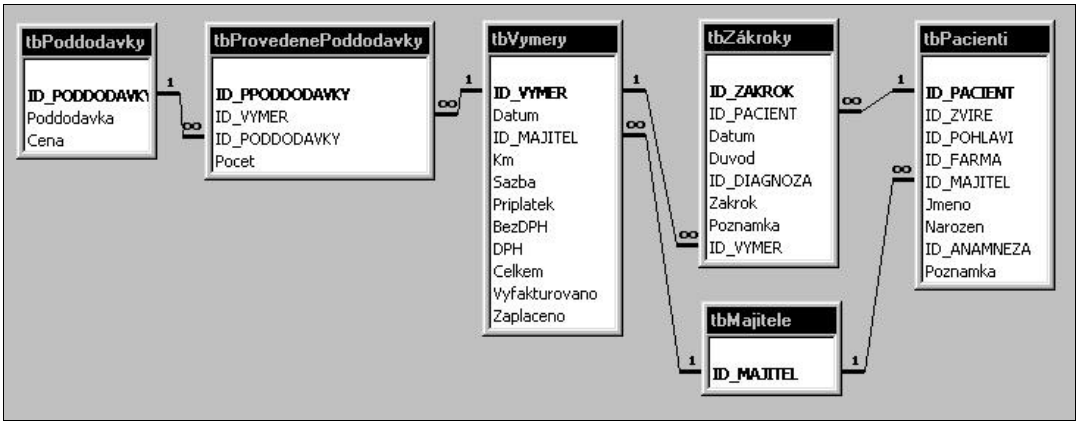
Atributy Km, Sazba a Priplatek souvisí s vykázáním cestovného. Částka, která se přičte ke konečné sumě výměru se rovná Km * Sazba + Priplatek.

Atributy BezDPH, DPH a Celkem souvisí s peněžní částkou. Tyto částky jsou automaticky vypočítávány programem. První dvě položky tvoří sumu cestovného, vykázaných poddodávek provedených úkonů a podaných léčiv. Celková částka je pak součtem částky bez DPH a DPH.

Tabulka obsahuje ještě atributy Vyfakturovano a Zaplaceno, které mají datový typ Ano/Ne. Jde spíše o informativní pole, které jsou využívány pro sestavy dlužníků a seznamu vypsaných faktur. Tyto atributy by také mohly sloužit pro informaci, zda byly data odeslána jinému datovému systému například účetnictví.

S tabulkou tbVymery úzce souvisí tabulka tbPoddodavky. Z této tabulky může uživatel vybrat poddodávky do vykázaného výměru. Poddodávky přiřazené ke konkrétnímu výměru jsou zahrnuty v tabulce tbProvedenéPoddodávky, kde jsou atributy ID_VYMER a ID_PODDODAVKY.

Tabulka tbParametry obsahuje údaje související s během aplikace a iniciály veterinární stanice nebo uživatele programu. Údaje jsou převážně využívány pro výpočet výměru a pro doplnění údajů v záhlaví faktury a tiskovou reprezentaci stanice. Tabulka má pouze jeden záznam pro všechny atributy a další záznamy nejsou uživateli zpřístupněny.



Obr. 8. Relace související s tabulkou tbVymery

5 APLIKACE UŽIVATELSKÉHO ROZHRANÍ

5.1 Model uživatelského rozhraní

Hlavní filozofií architektury uživatelského rozhraní je struktura aplikace typu MDI (Multiple Document Interface), kde aplikace obsahuje hlavní okno a z něho je spouštěno několik synovských oken. Pomocí tohoto modelu se uživateli po spuštění aplikace spustí hlavní přepínací panel, ve kterém můžeme volit další synovská okna. Tato podokna jsou pak svázaná s daty, se kterými bude uživatel pracovat.[8]

Vývoj uživatelského rozhraní typu MDI plně podporuje databázový systém Microsoft Access, proto, je tento použit i v této aplikaci. Microsoft Access nabízí průvodce, pomoci kterých můžeme snadno vytvořit formuláře. Navíc se zde nabízí snadná implementace kódu Visual Basicu.

Nevýhodou tohoto typu rozhraní se může jevit dojem nepřehlednosti. Uživatel totiž může spustit několik oken současně a na ploše obrazovky se pak okna mezi sebou překrývají. Není tedy přesně zřetelné, se kterým oknem se aktuálně pracuje. Tento typ aplikace postrádá tlačítko Uložit, proto může být uživatel někdy zmaten, jestli zapsaná data jsou potvrzena nebo ne.

Nicméně vzhledem k tomu, že jednotlivé tabulky si vyžadují jiný prostor pracovní plochy k zpracování dat, jeví se model MDI z hlediska kompaktnosti užitečný. Aplikace totiž kromě klasických formulářů obsahuje i formuláře zobrazené pomocí datového listu jako je tomu podobně v tabulkách. Tyto tabulky jsou v aplikaci použity pro lepší přehled uživatele v již uložených záznamech a tyto záznamy slouží i k výběru záznamů jiných tabulek související s vybraným záznamem.

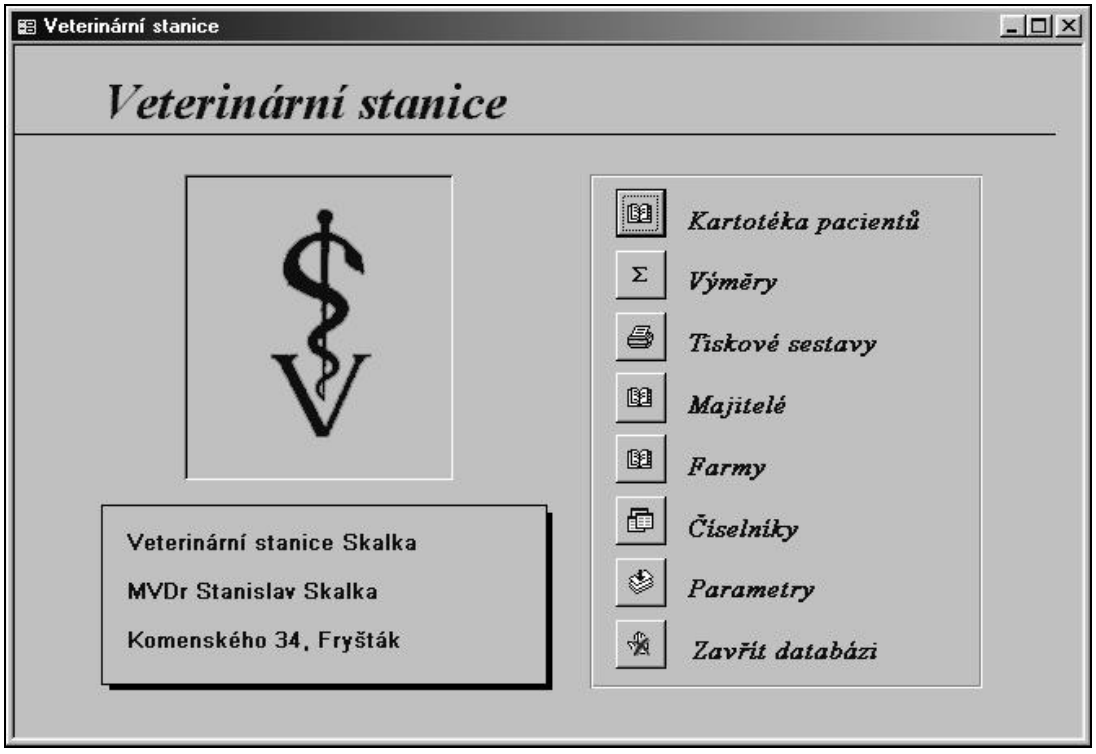
5.2 Hlavní přepínací panel

Po spuštění aplikace se zobrazí hlavní přepínací panel, který obsahuje tlačítka pro spuštění dalších synovských oken, která obsahují formuláře svázané s potřebnými daty. Tato tlačítka jsou obsloužena událostními procedurami ve Visual Basicu, stejně jako v dalších formulářích aplikace. Programový kód pak některé procedury odkazuje na spuštění manipulačních dotazů.

Panel ještě obsahuje informativní data o uživateli programu, která ovšem nelze z tohoto místa měnit. Ke změně těchto dat musí uživatel otevřít formulář frmParametry, kde kromě osobních údajů uživatele lze i měnit další nastavení funkce aplikace spojené s výpočty výměrů.

Pomocí hlavního panelu lze ještě zobrazit dva jakoby podpanely. První panel obsahuje seznam nabízených tiskových sestav. Tyto tiskové sestavy vytvářejí užitečné a přehledné souhrny dat databáze. Tiskové sestavy slouží pouze k nahlédnutí na poskytovaná data a jejich tisk, nikoliv na jejich úpravy. Tyto úpravy se mohou přehledně provádět v číselnících.

Číselníky slouží k editaci tabulek, jejichž struktura je jednoduchá a zpravidla obsahuje jen několik atributů, především název položky, popřípadě číselný kód, pokud není generován automaticky. Zajímavým číselníkem je tabulka léčiv a tabulka úkonů. Ty jsou totiž rozděleny do skupin, takže pokud se uživatel chce dostat k tabulce léčiv po kliknutí na číselník léčiv se mu napřed objeví tabulka skupin léčiv. Zde buď zapíše novou skupinu léčiv, nebo kliknutím na začátek řádku vybere příslušnou skupinu. Potom se teprve zobrazí tabulka léčiv, ale filtrovaná pro vybranou skupinu. Podobně se pracuje s tabulkou úkonů, zde jsou však skupiny rozděleny ještě navíc do oddílů.



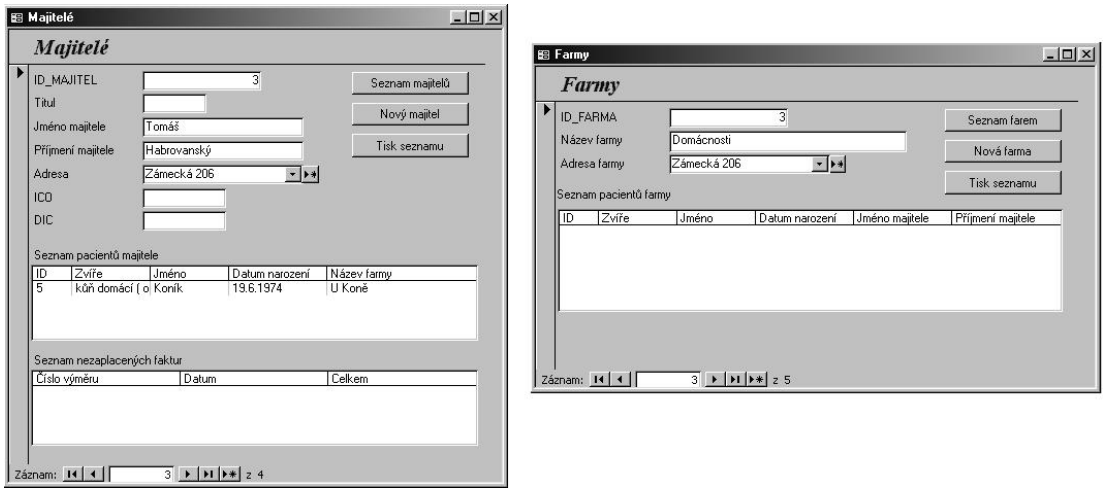
Obr. 9. Hlavní přepínací panel

5.3 Číselníky majitelé a farmy

Tyto číselníky jsou pro svou složitější strukturu implementovány do formuláře. Po otevření tohoto formuláře jsou data nabízena ve formulářovém zobrazení. Ve formuláři jsou navíc uvedena některá data související s příslušnými tabulkami.

Ve formuláři pro majitele jsou zobrazeny seznamy pacientů náležící k právě editovanému majiteli a také je zde zobrazen seznam nezaplacených faktur. Data v těchto seznamech však nelze editovat. Pro editaci těchto dat je nutné otevřít příslušný formulář.

Ve formuláři pro farmy jsou rovněž zobrazováni pacienti související s právě zobrazeným záznamem. Aplikace by dokonce umožňovala zobrazit farmy související s majitelem přes související pacienty. Tato modifikace by však nemusela být přehledná, protože na jedné farmě mohou být pacienti náležící několika majitelům. Proto je od této nabídky upuštěno.



Obr. 10. Formuláře Majitelé a Farmy

5.4 Formuláře pro kartotéku pacientů

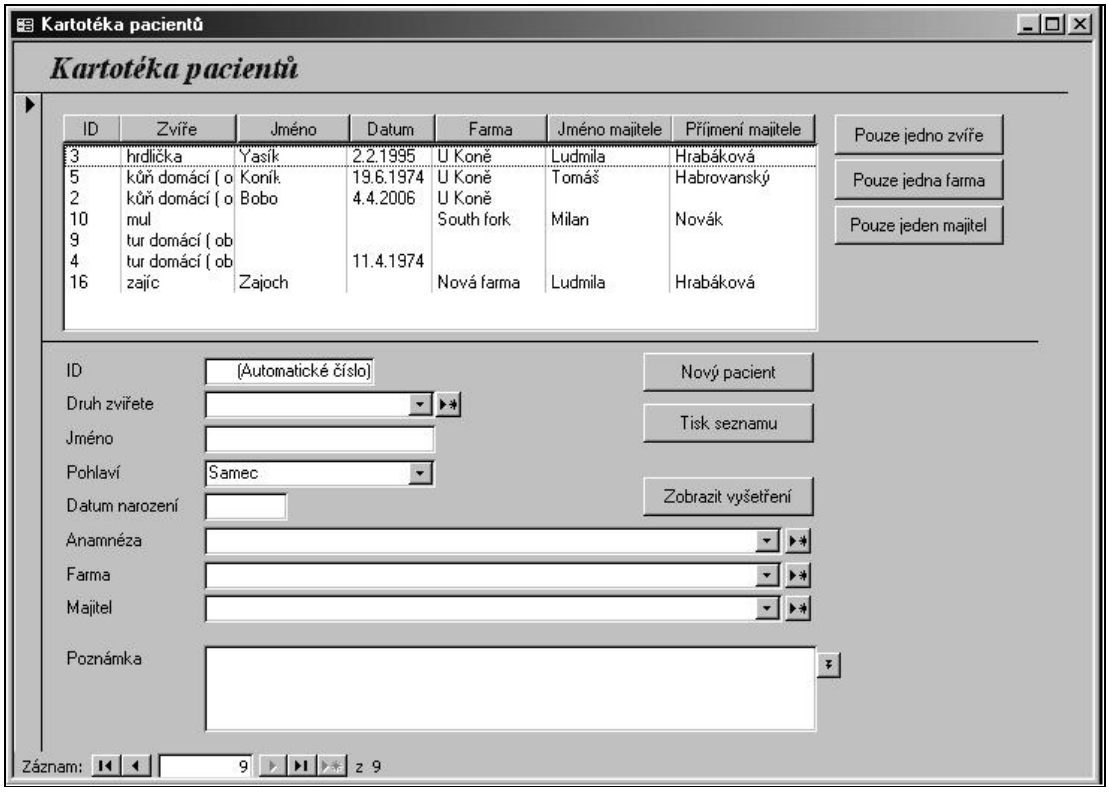
Jak již bylo výše uvedeno kartotéka pacientů se skládá ze dvou částí, ze seznamu pacientů a z vyšetření všech pacientů. Z toho důvodu jsou vytvořeny dva formuláře pro obsluhu dat kartotéky pacientů.

5.4.1 Formulář pacientů

První formulář frmPacienti obsahuje prvky pro zápis obecných zdravotních údajů a tělesných parametrů pacienta. Ovládací prvky formuláře jsou svázány s tabulkou tbPacienti. Položky záznamu Jméno a Datum narození musí uživatel zadat z klávesnice, položky Druh zvířete, Pohlaví, Anamnéza,Farma, Majitel musí uživatel zadat z příslušných tabulek, na které jsou vytvořené relace. U těchto položek je ještě přidáno ovládací tlačítko, kterým může uživatel rozbalit příslušný formulář položky a editovat její data. Většinou tato funkce slouží pro zapsání nového prvku související tabulky.

Zvláštním prvkem pak je položka Poznámka. Tu může uživatel buď vybrat z tabulky poznámek nebo zapsat jinou poznámku neuvedenou v seznamu. Z tohoto důvodu není vytvořena relace mezi tabulkou s poznámkami a tabulkou pacientů.

Pro lepší přehled je formulář obohacen seznamem všech pacientů. Aby byl výběr pacienta ze seznamu efektivnější je každý sloupec seznamu opatřen tlačítkem pro seřazení pacientů podle právě zvoleného atributu. Navíc je ještě možné pacienty filtrovat podle dalších atributů: podle vybraného majitele, farmy nebo druhu zvířete.



Obr. 11. Formulář pro zápis pacientů

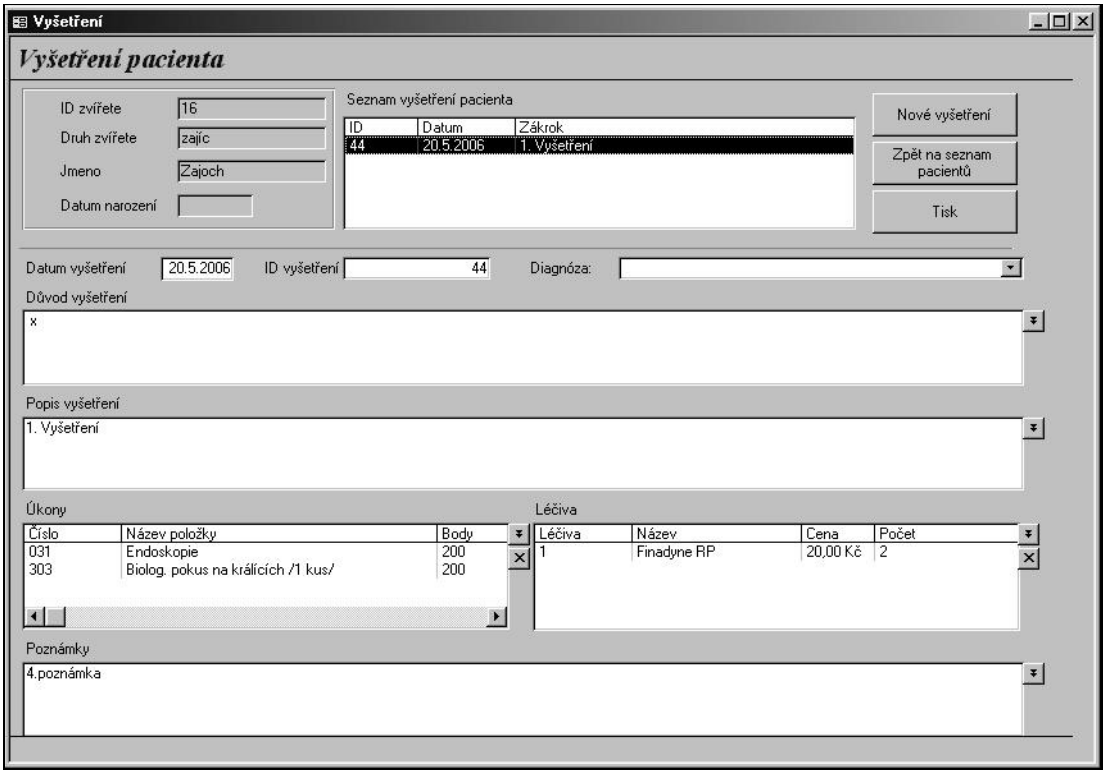
5.4.2 Formulář vyšetření

Ovládacím tlačítkem Zobrazit vyšetření nebo dvojitým kliknutím na seznam pacientů lze rozvinout formulář frmZakroky pro zápis jednotlivých vyšetření pacienta. Vzhledem k tomu, že se veškerá data ve formuláři vztahují k vybranému pacientovi, měl by být tento formulář z hlediska zachování integrity dat možný otevřít pouze z formuláře frmPacienti. Díky tomu se tak stává závislým podformulářem formuláře pacientů.

Formulář obsahuje seznam provedených vyšetření vybraného pacienta. Uživatel může vybrat již zapsané vyšetření, nebo vytvořit nové. Seznam vyšetření je v návrhovém prostředí stejně jako ve formuláři frmPacienti vytvořen pomocí komponenty Seznam a její událostní procedury jsou ovládány kódem ve Visual Basicu. Microsoft Access však nabízí ještě jedno řešení: Pomocí průvodce pro podformuláře lze vložit do formuláře podformulář a ten pak svázat se záznamy zobrazené v hlavním formuláři. Kliknutím na záznam podformuláře se zobrazí položky hlavního formuláře související s tímto záznamem. Protože formulář obsahuje ještě další související tabulky není pro složitost tento způsob implementace v aplikaci použit.

S vybraným vyšetřením pacienta totiž ještě souvisí tabulka podaných léčiv a tabulka provedených úkonů. Tyto tabulky jsou se záznamy vyšetření v relaci N:N. Více záznamů vyšetření může obsahovat stejné úkony a jeden záznam může obsahovat více úkonů. Výběr položek z tabulky úkonů probíhá tak, že se v příslušném místě formuláře rozbalí seznam úkonů a ten se pak kliknutím vybere do seznamu provedených úkonů. Výběrový dotaz pak zobrazí provedené úkony do příslušného seznamu ve formuláři.

Ve formuláři jsou položky Důvod vyšetření, Popis vyšetření a Poznámky. Tyto položky nejsou v relaci s příslušnými tabulkami. Je to ze stejného důvodu jako ve formuláři frmPacienti u položky Poznámky. Z příslušných tabulek není uživatel povinen vybírat jednotlivé záznamy, ale tyto záznamy jsou uživateli nabídnuty k zápisu. Uživatel sám rozhodne, zda-li použije věty ze seznamu, nebo je zapíše z klávesnice.



Obr. 12. Formulář pro zápis vyšetření pacienta

5.5 Formulář pro výměry

Formulář frmVymery slouží k vykazování lékařských vyšetření majiteli zvířat. Záznamy tohoto formuláře souvisí s tabulkou tbVymery. Obsahuje také ovládací tlačítko pomocí kterého lze formulář přepnout do zobrazení datového listu. Toto zobrazení nabízí uživateli

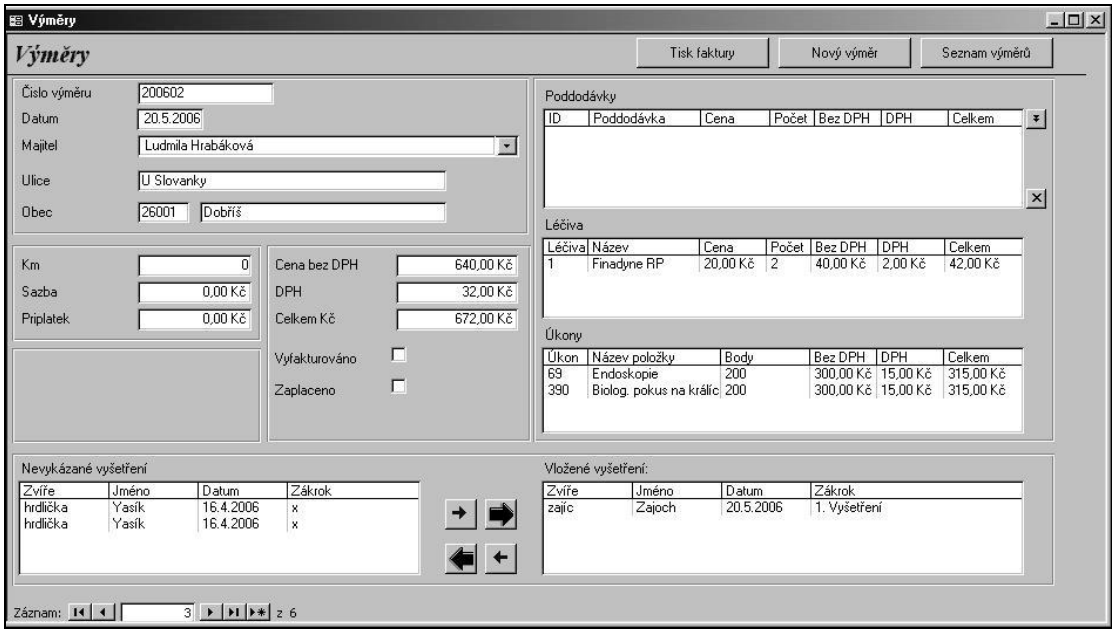
lepší přehled nad seznamem vykázaných výměrů. Navíc lze kliknutím na začátek řádku vybrat záznam, u kterého chce uživatel zobrazit podrobnosti nebo měnit data. Výběrem záznamu se pak formulář přepne zpět do formulářového zobrazení.

Při výběru nového záznamu musí uživatel nejprve zapsat číslo výměru nebo faktury. Pokud uživatel používá číselné vyjádření výměru, nabídne mu formulář automaticky číslo o jednu hodnotu vyšší než je nejvyšší uložené číslo výměru. Poté musí uživatel vybrat ze seznamu majitelů majitele, kterému chce výměr vykázat. Tím se automaticky vytvoří nabídka nevykázaných vyšetření související právě s vybraným majitelem. Seznam může obsahovat i více vyšetření jednoho pacienta a vyšetření může pocházet i z více vyšetřených zvířat.

Kliknutím na nabídnuté vyšetření nebo pomocí šipky se toto zařadí do seznamu vykázaných vyšetření související s tímto záznamem. Navíc se do příslušných seznamů ve formuláři objeví úkony a léky související s vybraným vyšetřením.

Uživatel může do výměru doplnit údaje o cestovném a vybrat z tabulky poddodávky. Při každé operaci uživatele se záznamem formuláře jsou pomocí událostní procedury ve Visual Basicu prováděny součty účetních položek provedených úkonů, podaných léčiv, podaných poddodávek a cestovného. Je vypočítána cena bez DPH, daň a celková suma.

Formulář obsahuje položky Vyfakturováno a Zaplaceno. Položka Vyfakturováno se automaticky nastaví na hodnotu Ano, pokud dojde ke stisknutí ovládacího tlačítka pro tisk faktury a naopak nastaveno na hodnotu Ne při jakékoliv operaci ovlivňující celkovou sumu výměru. Nicméně uživatel může tuto položku měnit podle vlastního uvážení a využívat ji pro další sestavy.



Obr. 13. Formulář pro vykazování výměrů

5.6 Zabezpečení databáze

Tato aplikace je ve své podobě určena jednomu uživateli. Z tohoto důvodu by se nabízel vytvořit jednoduchý zabezpečovací systém, který si při zpuštění databáze vyžádá jméno a heslo uživatele. Jednotlivé prvky objektů jako jsou formuláře, dotazy, sestavy, makra a programový kód by se mohl před nechtěným poškozením uživatelem chránit převedením souboru do formátu souboru MDE.

Vzhledem k tomu, že se struktura této aplikace do budoucna může rozrůst o možnost sdílet databázi více lékaři, kde lékaři buď budou mít své vlastní pacienty, nebo všichni uživatelé budou sdílet společnou kartotéku pacientů, je vhodné s tímto faktem počítat i v návrhu zabezpečení databáze.

V první fázi se vytvoří skupina administrátoři, kteří budou mít veškerá práva ke všem objektům aplikace. Další skupinou budou uživatelé, kde bude v podstatě zapsán jeden uživatel, která bude vlastnit aplikaci, i když by nic nebránilo do skupiny zapsat více lékařů. Této skupině pak budou přidělena práva pouze k přístupu k datům pro čtení, úpravy a zápis.

Aby si uživatel mohl změnit své vlastní jméno a heslo je potřeba vytvořit formulář, který je svázán s objekty kolekce Workspace User a Group. V těchto objektech budeme měnit

vlastnosti objektu třídy User.Password. metodou User.NewPasword. Aby nebylo možné toto heslo změnit neoprávněným uživatelem, obsahuje formulář ještě navíc původní přihlašovací jméno a heslo uživatele.

6 IMPLEMENTACE KÓDU VBA

Takřka všechna ovládací tlačítka požita ve formulářích jsou obsluhována událostními procedurami. Pro uživatele, který nikdy neprogramoval je výhodnější použít pro obsluhu ovládacích tlačítek makra, při jejichž tvorbě často pomohou průvodci pro vytváření maker. Nicméně jsem pro obsluhu procedur zvolil kód VBA, protože s makry si nemůžeme dovolit tolik, co s programovým kódem, kód je vestavěn přímo do formulářů, zatímco makra vytvářejí další samostatné databázové objekty.

Další výhodou programového kódu je, že můžeme podle potřeby nastavovat hodnoty argumentů jednotlivých funkcí, můžeme ošetřovat chyby vzniklé během chodu aplikace vlastním programovým kódem. Formulář, sestava i kód tak tvoří jeden nedílný celek, což zefektivňuje práci programátora. Také nám umožňuje pracovat s jednotlivými záznamy tabulky, což umožňuje dokonalou manipulaci s daty.

ZÁVĚR

Databázový systém je určen pro práci jednoho uživatele. Více uživatelů může používat databázi pouze pod stejným přihlašovacím jménem a heslem. Sdílejí tak společné pacienty, farmy i výměry. Systém lze rozšířit o tabulku uživatelů. Jednotliví uživatelé by tak mohli mít vlastní pacienty nebo společně sdílet s ostatními uživateli, fakturační výměry by souvisely pouze s právě přihlášeným uživatelem nebo skupinou uživatelů, pokud by patřily k jedné účetní jednotce. S tímto rozšířením se musí vytvořit patřičné tabulky uživatelů a formuláře pro přihlašování do systému. Také se musí upravit relace především v tabulkách pro pacienty a pro výměry k jednotlivým provedeným úkonům.

Pracovníci veterinární stanice musí také řešit otázku skladových zásob léčiv a jejich expirační dobu. Musí se také vytvářet výkazy služebních cest související s knihou jízd. Dále musí řešit náklady na provoz veterinární ordinace. Všechny tyto požadavky se navíc soustřeďují do účetnictví firmy. Proto by bylo vhodné nadstavit již vytvořený databázový systém o další moduly. Kartotéka pacientů, fakturační systém, správa lékárny, kniha jízd a účetnictví tak vytvoří komplexní systém pro správu dat veterinární stanice.

SEZNAM POUŽITÉ LITERATURY

[1] BÍLEK Martin. *MS Access 2.0 pro Windows.* 1. vyd. Praha: Grada Publishing, 1995. 168s. ISBN 80-7169-185-2.

[2] CASSEL Paul, EDDY Craig a PRICE John. *Nauč se sám Microsoft Access 2002 za 21 dní.* 1. vyd. Sams Publishing, 2002. 608 s. ISBN 80-86497-33-X.

[3] FEDDEMA Helen Bell. *Mistrovství v MS Access 2002: Access 2002 do nejmenších podrobností a do posledního detailu.* 1. vyd. Praha: Computer Press, 2002. ISBN 80-7226-725-6.

[4] KRAS Pavel. *Programování v MS Office v kostce VISUAL BASIC pro Excel a Access.* 1. vyd. Havlíčkův Brod: Fragment, 2000. 160 s. ISBN 80-7200-419-0.

[5] LACKO Luboslav. *Databáze: datové sklady, OLAP a dolování dat s příklady v Microsoft SQL Serveru a Oracle.* 1. vyd. Brno: Computer Press, 2003. 488 s. ISBN 80-7226-969-0.

[6] PELIKÁNOVÁ Lucie, ČIHÁK Jan, KNEJPLOVÁ Lucie. *VISUAL BASIC sbírka řešených příkladů.* 1. vyd. Praha: BEN, 1998. 208 s. ISBN 80-86056-40-6.

[7] POKORNÝ Jan. *Visual Basic pro aplikace Accessu.* 1. vyd. České Budějovice: KOPP, 1997. 176 s. ISBN 80-85828-80-4.

[8] RIORDAN Rebecca M. *Vytváříme relační databázové aplikace.* 1. vyd. Praha: Computer Press, 2000. 280 s. ISBN 80-7226-360-9.

[9] RUD Ollivia Parr. *Data mining. Praktický průvodce dolováním dat pro efektivní prodej, cílený marketing a popisu zákazníků (CRM).* 1. vyd. Brno: Computer Press. 322 s. ISBN 80-7226-577-6.

[10] STEPHENS Ryan K., PLEW Ronald R. *Naučte se SQL za 21*. dní. 1. vyd. Brno: Computer Press, 2004. 574 s. ISBN 80-722-6870-8.

[11] ŠEREŠA Ľubor, MIČOVSKÝ Aleš, ČERVEŇ Juraj. *Datové modelování v příkladech.* 1. vyd. Praha: Grada Publishing, 2001. 152s. ISBN 80-247-0049-2.

SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK

1NF	První normální forma
2NF	Druhá normální forma
3NF	Třetí normální forma
DAO	Data Access Objects
DBE	Database Engine
MDI	Multiple Document Interface
MS	Microsoft
NFNF	Nultá normální forma
SDI	Single Document Interface
VBA	Visual Basic for Applicatoins

SEZNAM OBRÁZKŮ

SEZNAM TABULEK

SEZNAM PŘÍLOH

PŘÍLOHA P I: TABULKY DATABÁZE

Tabulka	Atribut	Datový typ	Relace s tabulkami	Typ relace	Prim klíč
tbAdresy	ID_ADRESA	Autom. číslo	TbMajitele tbFarmy tbParmetry	1 1 1	Ano
	Ulice	Text			
	Obec	Text			
	PSC	Text			
tbAnamnezy	ID_ANAMNEZA	Autom. číslo	tbPacienti	1	Ano
	Anamneza	Text			
tbDiagnozy	ID_DIAGNOZA	Autom. číslo	tbZakroky	1	Ano
	Diagnoza	Text			
tbDruhyZvirat	ID_ZVIRE	Text	tbPacienti	1	Ano
	Zvire	Text			
tbDuvody	ID_DUVOD	Autom. číslo			Ano
	Duvod	Text			
tbFarmy	ID_FARMA	Autom. číslo	tbPacienti	1	Ano
	Nazev	Text			
	ID_ADRESA	Číslo	tbAdresy	N	

Tabulka	Atribut	Datový typ	Relace s tabulkami	Typ relace	Prim klíč
tbLeciva	ID_LECIVA	Text	tbPodanaLeciva	1	Ano
	Nazev	Text			
	ID_SKUPINA	Text	tbSkupinyLeciv	N	
	Cena	Měna			
	DPH	Procento			
tbMajitele	ID_MAJITEL	Autom. číslo	tbPacienti tbVymery	1 1	Ano
	Titul	Text			
	Jmeno	Text			
	Prijmeni	Text			
	Nazev	Text			
	ICO	Text			
	DIC	Text			
	ID_ADRESA	Číslo	tbAdresy	N	
tbOddilyUkonu	ID_ODDIL	Text	tbSkupinyUkonu	1	Ano
	Oddil	Text			
tbPacienti	ID_PACIENT	Autom. číslo	tbZakroky	1	Ano
	ID_ZVIRE	Číslo	tbDruhyZvirat	N	
	ID_POHLAVI	Číslo	tbPohlavi	N	
	ID_FARMA	Číslo	tbFarmy	N	
	ID_MAJITEL	Číslo	tbFarmy	N	

Tabulka	Atribut	Datový typ	Relace s tabulkami	Typ relace	Prim klíč
tbPacienti	Jmeno	Text			
	Narozen	Datum			
	ID_ANAMNEZA	Číslo	tbAnamnezy	N	
	Poznamka	Memo			
tbParametry	ID_PARAMETR	Autom. číslo			Ano
	Body	Měna			
	DPHukony	Procento			
	DPHcestovne	Procento			
	DPHpriplatky	Procento			
	DPHpoddodavky	Procento			
	ZaokrouleniVymeru	Desetinné číslo			
	ZaokrouhleniDPH	Desetinné číslo			
	Titul	Text			
	Jmeno	Text			
	Prijmeni	Text			
	ID_ADRESA	Číslo	tbAdresy	N	
	ICO	Text			
	DIC	text			
	Nazev	Text			
	Banka	Text			

Tabulka	Atribut	Datový typ	Relace s tabulkami	Typ relace	Prim klíč
tbParametry	Ucet	Text			
	Telefon	Text			
tbPodanaLeciva	ID_PLECIVO	Autom. číslo			Ano
	ID_ZAKROK	Číslo	tbZakroky	N	
	ID_LECIVO	Text	tbLeciva	N	
	Počet	Číslo			
tbPoddodavky	ID_PODDODAVKA	Autom. číslo	tbProvedene Poddodavky	1	Ano
	Poddodavka	Text			
	Cena	Měna			
tbPohlavi	ID_POHLAVI	Autom. číslo	tbPacienti	1	Ano
	Pohlavi	Text			
tbPoznamky	ID_POZNAMKY	Autom. číslo			Ano
	Poznamka	Memo			
tbProvedene Poddodavky	ID_PPODDODAVKA	Autom. číslo			Ano
	ID_PODDODAVKA	Číslo	tbPoddodavky	N	
	ID_VYMER	Text	tbVymery	N	
	Počet	Číslo			

Tabulka	Atribut	Datový typ	Relace s tabulkami	Typ relace	Prim klíč
tbProvedeneUkony	ID_PUKON	Autom. číslo			Ano
	ID_UKON	Číslo	tbUkony	N	
	ID_ZAKROK	Číslo	tbZakroky	N	
tbSkupinyLeciv	ID_SKUPINA	Text	tbLeciva	1	Ano
	Skupina	Text			
tbSkupinyUkonu	ID_SKUPINA	Autom. číslo	tbUkony	N	Ano
	ID_ODDIL	Text	tbOddilyUkonu	N	
	Cislo	Text			
	Skupina	Text			
tbUkony	ID_UKON	Autom. číslo	tbProvedeneUkony	1	
	ID_SKUPINA		tbSkupinyUkonu	N	
	Cislo	Text			
	Nazev	Text			
	Body	Desetinné číslo			
tbVymery	ID_VYMER	Text	tbZakroky	1	Ano
	Datum	Datum			
	ID_MAJITEL	Číslo	tbMajitele	N	
	Km	Desetinné číslo			
	Sazba	Měna			

Tabulka	Atribut	Datový typ	Relace s tabulkami	Typ relace	Prim klíč
tbVymery	Priplatek	Měna			
	BezDPH	Měna			
	DPH	Měna			
	Celkem	Měna			
	Vyfakturovano	Ano/Ne			
	Zaplaceno	Ano/Ne			
tbVysetreni	ID_VYSETRENI	Autom. číslo			Ano
	Vysetreni	Memo			
tbZakroky	ID_ZAKROK	Autom. číslo	tbProvedenyUkony tbPodanaLeciva	1 1	Ano
	ID_PACIENT	Číslo	tbPacienti	N	
	Datum	Datum			
	Duvod	Memo			
	ID_DIAGNOZA	Číslo	tbDiagnozy	N	
	Zakrok	Memo			
	Poznamka	Memo			
	ID_VYMER	Text	tbVymery	N	

PŘÍLOHA P II: FORMULÁŘE UŽIVATELSKÉHO ROZHRANÍ

Formulář	Popis funkce	Zdroj záznamů	Typ formuláře
frmAdresy	Číselník adres	tbAdresy	Datový list
frmAnamnezy	Číselník anamnéz	tbAnamnezy	Datový list
frmCiselniky	Přepínací panel se seznamem číselníků		Přepínací panel
frmDiagnozy	Číselník diagnóz	tbDiagnozy	Datový list
frmDruhyZvirat	Číselník druhů zvířat	tbDruhyZvířat	Datový list
frmDůvody	Číselník důvodů vyšetření pacienta	tbDuvody	Datový list
frmFarmy	Číselník farem se seznamem souvisejících pacientů	tbFarmy	Formulář Datový list
frmLeciva	Číselník léčiv	tbLeciva	Datový list
frmMajitele	Číselník farem se seznamem souvisejících pacientů a seznamem nezaplacených faktur	tbMajitele	Formulář Datový list
frmOddilyUkonu	Číselník oddílů úkonů	tbOddilyUkonu	Datový list
frmPacienti	Formulář pro výběr a zápis pacienta kartotéky pacientů	tbPacienti	Formulář
frmPanel	Hlavní přepínací panel, úvodní formulář uživatelského rozhraní	tbParametry	Přepínací panel
frmParametry	Editace parametrů pro výpočet výměrů a osobních dat uživatele	tbParametry	Formulář

Formulář	Popis funkce	Zdroj záznamů	Typ formuláře
frmPodanaLeciva	Formulář pro zápis podaných léčiv vybraných ve formuláři pro vyšetření	tbPodanaLaciva	Formulář
frmPoddodavky	Číselník poddodávek	tbPoddodávky	Datový list
frmPoznamky	Číselník poznámek	tbPoznamky	Datový list
frmProvedenePoddodavky	Formulář pro zápis provedených poddodávek vybraných ve formuláři pro výměry	tbProvedenePoddodavky	Formulář
frmProvedeneUkony	Formulář pro zápis provedených úkonů vybraných ve formuláři pro vyšetření	tbProvedeneUkony	Formulář
frmSkupinyLeciv	Číselník skupin léčiv	tbSkupinyLeciv	Datový list
frmSkupinyUkonu	Číselník skupin úkonů	tbSkupinyLeciv	Datový list
frmTisk	Přepínací panel pro výběr tiskové sestavy		Přepínací panel
frmUkony	Číselník úkonů	tbUkony	Datový list
frmVymery	Formulář pro vykázání výměrů a výpočet částky pro zaplacení	tbVymery	Formulář Datový list
frmVysetreni	Číselník popisu vyšetření pacienta	tbVysetreni	Datový list
frmZakroky	Formulář pro zápis vyšetření pacienta související s vybraným pacientem ve formuláři pro pacienty	tbZakroky	Formulář

PŘÍLOHA P III: MANIPULAČNÍ DOTAZY DATABÁZE

Dotaz	Popis	Objekty použití	Typ dotazu
qryAdresaStanice	Vypíše adresu veterinární stanice	frmPanel	Výběrový
qryDluznici	Seznam majitelů s nezaplacenými fakturami	rptDluznici	Výběrový
qryLecivaSVymerem	Seznam podaných léčiv související s vybraným výměrem	frmVymery rptFaktura	Výběrový
qryLecivaSVymeremCelkem	Suma částky podaných léčiv související s vybraným výměrem	frmVymery rptFaktura	Výběrový
qryMajiteleSAdresou	Vypíše vybraného majitele s atributy jeho adresy	rptMajitele	Výběrový
qryNezaplaceneFaktury	Seznam nezaplacených faktur související s vybraným majitelem	frmMajitele rptMajitele	Výběrový
qryOdeberJedenZakrok	Odebere z vybraného výměru jeden vybraný zákrok	frmVymery	Aktualizační
qryOdeberVsechnyZakroky	Odebere z vybraného výměru všechny související zákroky	frmVymery	Aktualizační
qryOdstranitPoddodavku	Odebere z vybraného výměru související vybranou poddodávku	frmVymery	Aktualizační
qryPacientiPodleData	Seřadí seznam pacientů podle data narození	frmPacienti	Výběrový

Dotaz	Popis	Objekty použití	Typ dotazu
qryPacientiPodleFarmy	Seřadí seznam pacientů podle farem	frmPacienti rptPacientiFarem	Výběrový
qryPacientiPodleID	Seřadí seznam pacientů podle ID	frmPacienti	Výběrový
qryPacientiPodleJmena	Seřadí seznam pacientů podle jména	frmPacienti	Výběrový
qryPacientiPodleJmenaMajitele	Seřadí seznam pacientů podle jména majitele	frmPacienti	Výběrový
qryPacientiPodleNarozeni	Seřadí seznam pacientů podle data narození	frmPacienti	Výběrový
qryPacientiPodleprijmeni	Seřadí seznam pacientů podle příjmení majitele	frmPacienti rptPacientiMajitele	Výběrový
qryPacientiPodleZvire	Seřadí seznam pacientů podle druhu zvířat	frmPacienti	Výběrový
qryPacientiPouzeFarma	Seznam pacientů souvisejících s vybranou farmou	frmPacienti	Výběrový
qryPacientiPouzeMajitel	Seznam pacientů souvisejících s vybraným majitelem	frmPacienti	Výběrový
qyrPacientiPouzeZvire	Seznam pacientů souvisejících s vybraným druhem zvířete	frmPacienti	Výběrový
qryParametry	Výpis vybraných parametrů	frmVymery frmPanel rptFaktura	Výběrový

Dotaz	Popis	Objekty použití	Typ dotazu
qryPodanaLeciva	Seznam léčiv související s vybraným vyšetřením	frmZakroky rptZakroky	Výběrový
qryPoddodavkyVymeru	Seznam poddodávek související s vybraným výměrem	frmVymery frmFaktura	Výběrový
qryPoddoavkyVymeruCelkem	Suma částky poddodávek související s vybraným výměrem	frmVymery frmFaktura	Výběrový
qryProvedeneUkony	Seznam úkonů související s vybraným vyšetřením	frmZakroky rptZakroky	Výběrový
qrySkupinyLeciv	Seznam léčiv související s vybranou skupinou léčiv	frmZakroky rptZakroky	Výběrový
qrySkupinyOddilu	Seznam skupin úkonů související s vybraným oddílem úkonů	frmZakroky	Výběrový
qrySkupinyUkonu	Seznam úkonů související s vybranou skupinou úkonů	frmZakroky rptZakroky	Výběrový
qrySmazLeciva	Vymaže vybrané léčivo ze seznamu podaných léčiv související s vybraným vyšetřením	frmZakroky	Odstraňovací
qrySmazUkony	Vymaže vybrané úkony ze seznamu provedených úkonů související s vybraným vyšetřením	frmZakroky	Odstraňovací
qryTelefonniSeznam	Telefonní seznam majitelů	rptTelefonniSeznam	Výběrový

Dotaz	Popis	Objekty použití	Typ dotazu
qryUkonySVymerem	Seznam provedených úkonů související s vybraným výměrem	frmVymery rptFaktura	Výběrový
qryUkonySVymeremCelkem	Suma částky provedených úkonů související s vybraným výměrem	frmVymery rptFaktura	Výběrový
qryUlozUkon	Přidá úkon do tabulky provedených úkonů	frmZakroky	Přidávací
qryVyberJedenZákrok	Přidá jedno vybrané vyšetření do vybraného výměru	frmVymery	Aktualizační
qryVyberVsechnyZakroky	Zapíše všechny zákroky související s vybraným majitelem do vybraného výměru	frmVymery	Aktualizační
qryVyberZakroku	Seznam zákroků související s vybraným pacientem	frmZakroky	Výběrový
qryVymerMax	Vybere maximální hodnotu čísla výměru	frmVymery	Výběrový
qryVysetreni BezVymeru	Seznam nevykázaných vyšetření související s vybraným majitelem výměru	frmVymery	Výběrový
qryVysetreniSVymerem	Seznam vykázaných vyšetření související s vybraným výměrem	frmVymery rptFaktura	Výběrový
qryZakrok	Atributy záznamu související s vybraným vyšetřením	rptZakroky	Výběrový

PŘÍLOHA P IV: TISKOVÉ SESTAVY

Tisková sestava	Popis	Zdroj záznamů	Podsestavy
Léčiva:	Podsestava faktury Seznam léčiv související s vyšetřením, která jsou vykázaná ve faktuře	qryVysetreniSVymerem	
Poddodavky	Podsestava faktury Seznam poddodávek vykázaných ve faktuře	qryProvedenePoddodavky	
Podaná léčiva	Podsestava vyšetření, zobrazuje podaná léčiva související se zobrazeným vyšetřením	qryPodanaLeciva	
Provedené úkony	Podsestava vyšetření, zobrazuje úkony související se zobrazeným vyšetřením	qryProvedeneUkony	
rptAdresy	Seznam adres	tbAdresy	
rptDluznici	Seznam majitelů s nezaplacenými fakturami	qryDluznici	
rptDruhyZvirat	Seznam druhů zvířat	qryDruhyZvirat	
rptFaktura	Faktura vykázaného výměru	tbParametry frmVymery	Léčiva: Úkony: Poddodávky
rptFarmy	Seznam farem	tbFarmy	
rptLeciva	Seznam léčiv roztříděných	qrySkupinyLeciv	

Tisková sestava	Popis	Zdroj záznamů	Podsestavy
	do skupin		
rptMajitele	Seznam majitelů	tbMajitele	
rptPacientiFarem	Seznam farem se souvisejícími pacienty	qryPacientiPodleFarmy	
rptPacientiMajitelu	Seznam majitelů se souvisejícími pacienty	qryPacientiPodleMajitele	
rptPoddodavky	Seznam poddodávek	tbPoddodavky	
rptUkony	Seznam úkonů roztříděných do skupin a oddílů	qrySkupinyUkonu	
rptZakroky	Vybrané vyšetření pacienta včetně podaných léčiv a provedených úkonů	qryZakroky	Podaná léčiva Provedené úkony
Úkony:	Podsestava faktury Seznam úkonů související s vyšetřením, která jsou vykázaná ve faktuře	qryVysetreniSVymerem	

PŘÍLOHA P V: DATABÁZE SPRÁVY VETERINÁRNÍ STANICE – MS ACCESS

V přiloženém CD-ROM se nachází soubor aplikace VETERINÁRNÍ STANICE.MDB.