Česká zemědělská univerzita v Praze

Provozně ekonomická fakulta

Katedra informačního inženýrství

Bakalářská práce

DBS v informačním zajištění sportovních akcí

Kateřina Zapletalová

ČESKÁ ZEMĚDĚLSKÁ UNIVERZITA V PRAZE

Katedra informačního inženýrství

Provozně ekonomická fakulta

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

Zapletalová Kateřina

Informatika

Název práce

DBS v informačním zajištění sportovních akcí

Anglický název

DBS in the information security of sports events

Cíle práce

Bakalářská práce je tématicky zaměřená na využití db technologie v informačním zabezpečení sportovních akcí. Cílem práce je:

a) vymezit teoretické principy relačně databázové technologie v souvislosti s informačním zabezpečením sportovních aktivit,

b) zmapovat současnou úroveň využití db technologie v informačním zabezpečení sportovních akcí a identifikovat bariéry jejiho širšího uplatnění,

c) navrhnout odstranění identifikovaných bariér,

d) navrhovaná řešení ověřit a demonstrovat na konkrétním příkladu z praxe,

e) ověřené záležitosti zobecnit pro další možná použití.

Metodika

Použitá metodika zadané bakalářské práce bude založena na studiu a analýze dostupných informačních zdrojů. Navrhované řešení bude realizováno formou praktické aplikace, která bude respektovat identifikované bariéry. Na podkladě syntézy teoretických poznatků a dosažených výsledků budou formulovány závěry této bakalářské práce a následně zobecněny pro další možná použití.

Harmonogram zpracování

Vymezení teoretických principů relačně db technologie - předmět 1. zápočtu z BP za 2. ročník studia: 06/2012-09/2012.

Zmapování současné úrovně využívání db technologie v evidenci sportovních akcí a identifikace existujících bariér: 09/2012-11/2012

Navržení konkrétního řešení - předmět 1. zápočtu z BP za 3. ročník: 11/2012-01/2013.

Ověření navrženého řešení a jeho zobecnění - předmět 2. zápočtu z BP za 3. ročník: 02/2013-03/2013.

Rozsah textové části
40-50 stran

Klíčová slova
DBS, informační zabezpečení, SQL, datová normalizace, datová integrita

---

Doporučené zdroje informací

KRUCZEK, A.: Microsoft Access 2010. COMPUTER PRESS 2010. EAN: 9788025132890
LACKO, L.: 1001 tipů a triků pro SQL. COMPUTER PRESS 2011. EAN: 9788025130100
MOLINARO, A.: SQL kuchařka programátora. COMPUTER PRESS 2009. EAN:9788025126172

Vedoucí práce
Vostrovský Václav, doc. Ing., Ph.D.

Termín odevzdání
březen 2013

Ing. Martin Pelikán, Ph.D.
Vedoucí katedry

ČESKÁ ZEMĚDĚLSKÁ UNIVERZITA V PRAZE
Provozně ekonomická fakulta

prof. Ing. Jan Hron, DrSc., dr.h.c.
Děkan fakulty

V Praze dne 5.10.2012

Čestné prohlášení

Prohlašuji, že svou bakalářskou práci "DBS v informačním zajištění sportovních akcí" jsem vypracovala samostatně pod vedením vedoucího bakalářské práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou citovány v práci a uvedeny v seznamu literatury na konci práce. Jako autorka uvedené bakalářské práce dále prohlašuji, že jsem v souvislosti s jejím vytvořením neporušila autorská práva třetích osob.

V Praze dne 13.3.2013 _____

Poděkování

Ráda bych touto cestou poděkovala doc. Ing. Václavu Vostrovskému, Ph.D. za cenné rady, připomínky a metodické vedení práce.

DBS v informačním zajištění sportovních akcí

DBS in the information security of sports events

Souhrn

Bakalářská práce se zabývá problematikou uplatnění jazyka SQL v informačním zajištění sportovních akcí v prostředí Microsoft Access. V první části práce jsou vysvětleny principy relačních databází a její historie, dále pak pojmy. V této části je teoreticky popsán jazyk SQL, blíže příkaz SELECT, který je uplatňován v praktické části. V poslední kapitole jsou vymezeny datová integrita a datová normalizace.

V druhé části práce je vysvětlen současný stav evidování sportovních aktivit, basketbalu. Dle tématu je zde namodelován ER model krajské soutěže. Dále jsou zde demonstrovány praktické ukázky využití SQL jazyka. V neposlední řadě využití Microsoft Access pro tuto práci.

Summary

This bachelor thesis is focused on a usage of SQL language for information access of sport events through Microsoft Access. In the first part, there is an explanation of basic principles of relational databases and its history together with further concepts. In this part one may find the theoretical description of SQL language and more closely described SELECT command, which practical use is further described in the practical part of this thesis. The last part specifies the data integrity and normalization of data.

The second part of this thesis focuses on the current state of evidence of sport events, basketball in particular. In compliance with the topic there is prepared an ER model for regional competitions. Moreover, one may find there some demonstrations of practical examples of the usage of SQL language. Last but not least, there is also shown the use of Microsoft Access for this type of a work.

Klíčová slova: DBS, informační zabezpečení, SQL, datová normalizace, datová integrita

Keywords: DBS, information security, SQL, normalization of data, data integrity

Obsah:

1.Úvod

Pojem databáze je v dnešní době hojně používaný. Potřebu evidovat a shromažďovat informace mají lidé už odpradávna. Téměř každý si pod pojmem databáze představí soubor informací, dat, které spolu nějak souvisí jako seznam klientů společnosti, seznam studentů či záznamy o pacientech. Každý z nás používá různé databázové aplikace, aniž bychom si to uvědomovali.

Počátky databázového zpracování dat se počítají od 70.let 20.století. Bylo nutné vytvořit jazyk, sadu příkazů k ovládání databází. Tento jazyk se měl podobat co nejvíce přirozenámu jazyku, angličtině. Vznikl tak jazyk SEQUEL později přejmenován na SQL (Structured Query Language). SQL podporují všechny moderní databázové systémy. Oracle, MySQL, Microsoft SQL server nebo Microsoft Access.

Velmi rozšířený je právě program Microsoft Access, který byl vybrán pro zpracování této bakalářské práce. Hlavním kladným aspektem tohoto programu je ten, že práce s ním je jednoduchá a přehledná.

2.Cíl práce a metodika

2.1. Cíl práce

Bakalářská práce je tématicky zaměřená na využití db technologie v informačním zajištění sportovních akcí. Cílem práce je vymezit teoretické principy relačně databázové technologie v souvislosti s informačním zajištěním sportovních aktivit, zmapovat současnou úroveň využití db technologie v informačním zajištění sportovních akcí a identifikovat bariéry jejího širšího uplatnění, navrhnout odstranění těchto bariér. Navrhovaná řešení ověřit a demonstrovat na konkrétním příkladu z praxe, ověřené záležitosti zobecnit pro další možná použití.

2.2. Metodika

Použitá metodika zadané bakalářské práce bude založena na studiu a analýze dostupných informačních zdrojů. Navrhované řešení bude realizováno formou praktické aplikace, která bude respektovat identifikované bariéry. Budou použity techniky relačně databázového návrhu, datové normalizace a integrity dat. Na podkladě syntézy teoretických poznatků a dosažených výsledků budou formulovány závěry této bakalářské práce a následně zobecněny pro další možná použití.

3. Teoretická část

3.1. Od počátků

3.1.1. Edgar “Ted” Codd

Edgar Codd (23.srpna 1923 – 18.dubna 2003) byl britský matematik a počítačový vědec. Poté, co sloužil jako pilot v Royal Air Force během druhé světové války, začal svou kariéru v New Yorku jako matematický programátor. V roce 1960 začal s doktorským studiem na Michiganské universitě v Ann Arboru, která byla sponzorována prostřednictvím stipendií IBM.

Po dokončení doktorského studia v informatice v roce 1967 pokračoval v kariéře s IBM na Almaden Research Center v San Jose, v Kalifornii. Zde začal přemýšlet nad otázkou, jak nejlépe spravovat data v existujících počítačových systémech.

V červnu 1970 Codd publikoval dokument Relační model dat pro velké sdílené databanky. V roce 1976 mu bylo uděleno stipendium na IBM a v roce 1981 získal nejprestižnější ocennění v oblasti informatiky, AM Turing Award, pro jeho příspěvky k teorii relačních databázových systémů.[1]

Codd pokračoval v IBM až do roku 1984, kdy odešel do důchodu. V této době spolu se svým bývalým kolegou z IBM Datumem Chrisem tvořili Consulting Group. Další člen této skupiny byla matematička z IBM Sharon Weinberg, která se později stala druhou Coddovou manželkou. Codd vydal řadu 12 pravidel pro definování toho, co představuje relační databáze. Doklad o souladu s většinou těchto pravidel může být viděn ve většině implementací RDBMS dnes. (RDBMS je zkratka pro relační databázový systém, data jsou uspořádána v databázových tabulkách, polí a záznamech)

12 Coddových pravidel:

„1. Informační pravidlo:

Všechny informace v relační databázi jsou vyjádřeny explicitně na logické úrovni jediným způsobem - hodnotami v tabulkách.

2. Pravidlo jistoty:

Všechna data v relační databázi jsou zaručeně přístupná kombinací jména tabulky s hodnotami primárního klíče a jménem sloupce.

3. Systematické zpracování nulových hodnot:

Nulové hodnoty jsou plně podporovány relačním SŘBD pro reprezentaci informace, která není definována a to nezávisle na datovém typu.

4. Dynamický on-line katalog založený na relačním 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 apod.

6. Pravidlo vytvoření pohledů:

Všechny pohledy, které jsou teoreticky možné, jsou také systémem 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 změnách 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:

Jestliže má relační systém jazyk nízké úrovně, pak tato úroveň nemůže být použita k vytváření integritních omezení a je nutno vyjádřit se v relačním jazyce vyšší úrovně.“ [2]

3.1.2. První jazyk

Bylo nutné vytvořit sadu příkazů k ovládání těchto databází, jazyk, který by se tvořil syntakticky a byl by co nejblíže přirozenému jazyku (angličtina). Vznikl tak jazyk SEQUEL (Structured English Query Language), který byl později přejmenován na SQL (Structured Query Language).

3.1.3. DB2

DB2 je relační databázový systém vyvinutý společností IBM. Je první databázový produkt, který používá SQL jazyk. DB2 byl poprvé k dispozici na mainframu a později přešel do Unixu, Linuxu a Windows. Má schopnost zvládnout miliony transakcí a obrovských souborů dat.

3.1.4. Oracle

Oracle je relační databázový systém vyvinutý společností Oracle Corporation. Oracle RDBMS se používá v mnoha databázových aplikací na platformách sevarl OS, včetně Unix a Windows. Tento databázový systém byl původně vyvinut Larrym Ellisonem, Bobem Minerem a Edem Oatesem. Jejich společnost Software Development Laboratories byla založena v roce 1977, v roce 1983 byla přejmenována na Oracle Corporation. Oracle databáze byla první databáze v souladu s ANSI SQL standardu. Oracle databáze je dražší ve srovnání s MS SQL Servrem.[3]

3.2. Základní databázové pojmy

3.2.1. Entita

Entita je hlavní a nejdůležitější prvek. Je to existující prvek reálného světa (člověk, televize, zvíře a nebo jakýkoliv jiný předmět)

3.2.2. Atribut

Každá entita má vlastnost, charakteristiku (název, hmotnost, barva atd.)

3.2.3 Vazba

Vazba mezi entitami. Každá entita odpovídá určitým prvkům z reálného světa, a ty mají mezi sebou určitý vztah.

Vazba 1:1 – Tato vazba nastává tehdy, kdy jedna entita v první tabulce odpovídá právě jedné entitě v tabulce druhé. Např. člověk má právě jedno rodné číslo. Tato relace není tak častá, protože dvě tabulky, které se propojují touto vazbou, mají většinou hodně společného, proto se většinou slučují do jedné tabulky a tato relace se vypouští.

Vazba 1:N – Tato vazba nastává tehdy, kdy jedna entita odpovídá několika entitám v tabulce druhé. Např. člověk může vlastnit několik aut, ale auto vždy může mít jen jednoho vlastníka. Tato relace je nejpoužívanější.

Vazba M:N – Několik entit z jedné tabulky odpovídá několika entitám z druhé tabulky, toto je vazba M:N. Tato relace se sestavuje pomocí dvou relací 1:N a pomocné tabulky s primárními klíči.

3.3. Základní objekty databáze

3.3.1. Tabulka

Tabulky se skládají ze sloupců a z řádků, a tím jsou databázové tabulky velmi podobné těm běžným tabulkám. Řádek se nazývá záznam, kde jsou uloženy jednotlivé informace. Každý záznam se skládá z jednoho či více polí, které odpovídají počtu sloupců v tabulce. Každé pole může mít jiný datový typ (text, datum, číslo atd.)

3.3.2 Index

Index je objekt, ve kterém je uložený klíč, podle kterého jsou data z tabulek nějak uspořádaná. Tento klíč si určujeme na začátku vytváření tabulky. **Primární klíč** může v dané tabulce existovat pouze jeden, musí být jedinečný. Data ve sloupci, nad kterým je tento klíč, se nesmí opakovat a nesmí mít hodnotu NULL. **Cizí klíč,** těchto klíčů může být několik, pomocí těchto klíčů si zajišťujeme integritu dat a spojení tabulek relacemi (1:1, 1:N, M:N).

3.3.3 Pohled

Pohled si lze představit jako virtuální tabulku nad skutečnou tabulkou, ale na rozdíl od ní nelze v pohledu manipulovat s informacemi. Pohledem lze zajistit zabezpečení dat tím, že uživateli zpřístupníme jen data, která zpřístupnit chceme.

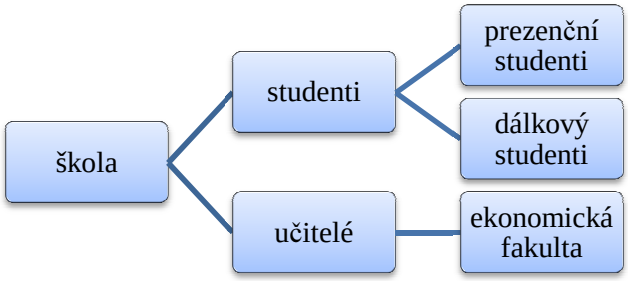
3.3.4 Triggery

„Spoušť“ je SQL procedura, která zahájí akci, když nastane událost (INSERT, DELETE nebo UPDATE). Například pokud je trigger definován pro DELETE na konkrétní tabulky, spoušť nastane vždy, když někdo odstraní řádek či řádky z tabulky. Triggery jsou používány k udržení referenční integrity dat.

3.4. Databázové modely

Existují základní tři databázové modely a to hierarchický, síťový a relační model dat. Nejnovější, nejpoužívanější a nejefektivnější model je relační, který odstraňuje chyby předešlých databázových modelů.

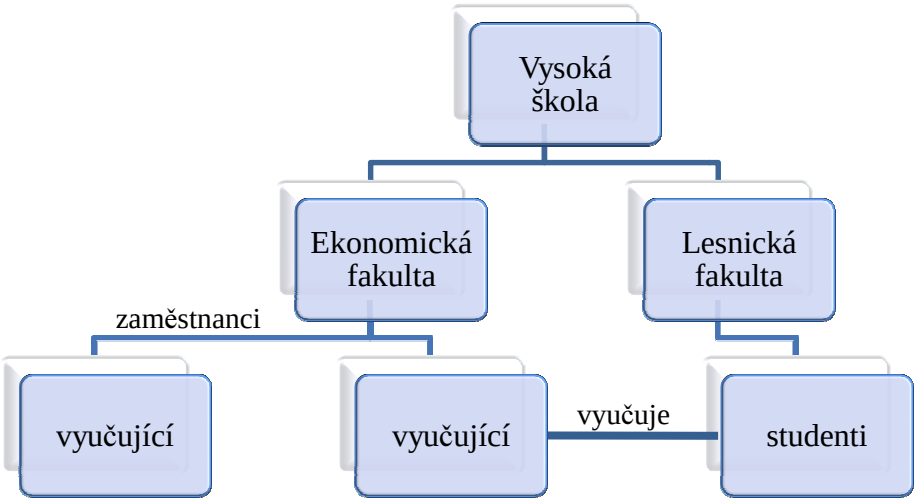
3.4.1 Hierarchický model



Obrázek č.1 Hierarchický model, zdroj:vlastní

Data jsou organizována do stromové struktury od kořene až po jednotlivé listy. Jednotlivé záznamy mají mezi sebou vztah typu rodič/potomek. Výhodou tohoto modelu je zajištění referenční integrity (zabezpečení). Záznamy v tabulce potomka jsou přímo napojeny na záznam tabulky rodič, při vymazání záznamu v tabulce rodič, bude smazán i propojený záznam v tabulce potomek. Nevýhodou tohoto modelu je možný výskyt redundantních dat.

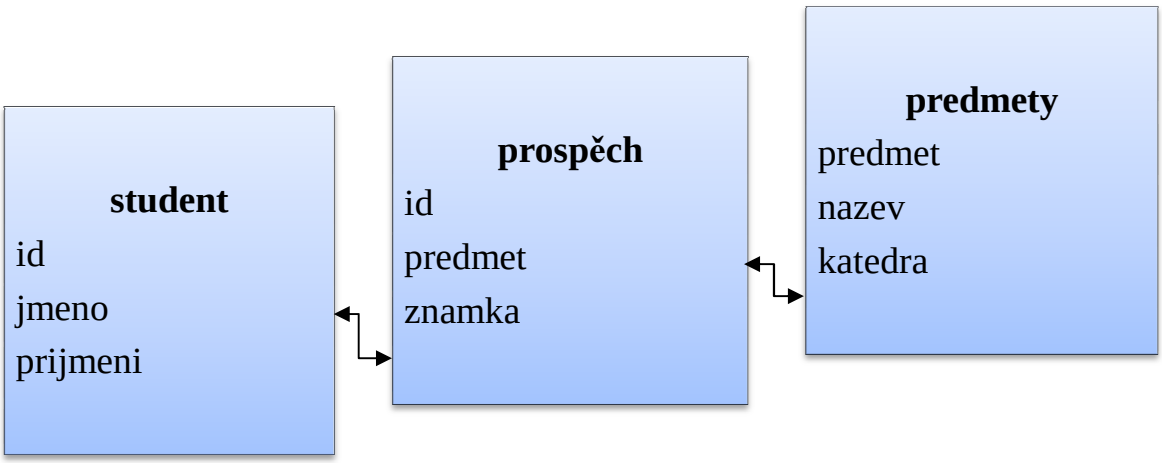
3.4.2. Síťový model



Obrázek č.2 Síťový model, zdroj:vlastní

Síťový model je pokus o vylepšení hierarchického modelu. Nedostatečná flexibilita a obtížná změna struktury je nevýhodou síťového modelu. Existují zde mnohanásobné vztahy nebo-li Sety, které propojují různé záznamy, což u předchozího modelu nešlo.

3.4.3. Relační model



Obrázek č.3 Relační model, zdroj:vlastní

V roce 1970 popsán Dr. Coddem. Data jsou uspořádána do tabulek, které mají sloupce a řádky, a jsou zde prováděny všechny databázové operace. Tabulky jsou vzájemně propojovány vazbami (relacemi) 1:1, 1:N a M:N.[4]

3.5. Jazyk SQL

SQL je zkratka pro Structured Query Language. Jazyk SQL se používá k vytváření, transformaci a načítání informací z RDBMS (relačních databázových systémů). SQL byl vyvinut během počátku 70. let v IBM. Většina relačních databázových systémů jako například MS SQL Server, Microsoft Access, Oracle, MySQL, DB2, Sybase, PostgreSQL a Informix používají SQL dotazování jazyk. I když je SQL definován jak ISO a ANSI existuje mnoho SQL implementací, které nejsou plně v souladu s těmito definicemi. Některé z těchto implementací jsou proprietární. Příklady těchto dialektů jsou například MS SQL Server specifická verze SQL tzv. T-SQL a Oracle verze SQL s názvem PL / SQL. SQL je deklarativní programovací jazyk navržený pro tvorbu a dotazování relačních systémů pro správu databází. Je to poměrně jednoduchý jazyk. Pomocí SQL lze vložit data do databázových tabulek, může upravovat data v existujících databázových tabulkách, lze odstranit data z databázových tabulek. SQL používá sadu příkazů pro manipulaci s daty v relačních databázích.

ZÁKLADNÍ PRAVIDLA PRO PRÁCI S SQL:

- Velká/Malá písmena (když je to tak definováno)
- Správné použití mezer
- Klíčová slova

VÝHODY:

- Velmi rychlý přístup
- Existuje několik programů volně k dispozici ke stažení, které jsou založeny na SQL jazyce
- Vysoká spolehlivost
- Jazyk je navržen tak, že ho mohou používat i amatéřští programátoři
- Jsou možné komplexní dotazy
- Nezávislý na určitých databázových programech

NEVÝHODY:

- Náklady na správu, udržování a zajišťování jsou vysoké
- I když je tento jazyk jednoduchý, musí se striktně dodržovat syntaxe příkazů
- Únavné zadávání příkazů oproti pohodlnějšímu „drag and drop“

3.5.1 Příkazy SQL

Tento jazyk je rozdělen do tří skupin.

Skupina první: Data Manipulation Language (DML) – manipulace s daty

- SELECT – vybírá data z databáze
- UPDATE – aktualizuje data v databázi
- DELETE – smaže data z databáze
- INSERT INTO – vloží nová data do databáze

Skupina druhá: Data Definition Language (DDL) – definice dat

- CREATE DATABASE – vytvoří novou databázi
- ALTER DATABASE – upravuje databázi
- CREATE TABLE – vytvoří novou tabulku
- ALTER TABLE – upravuje tabulku
- DROP TABLE – odstraní tabulku
- CREATE INDEX – vytvoří index
- DROP INDEX – odstraní index

Skupina třetí: Data Control Language (DCL) – řízení dat

-jsou to příkazy pro přidělování a odebírání práv (práva systémová a objektová)

Některé příkazy budou vysvětleny na názorném příkladu.

VATVOŘENÍ TABULKY:

- **CREATE TABLE**

-příkaz CREATE TABLE vytváří novou tabulku, v této tabulce musí být vytvořeny sloupce se správným datovým typem a rozsahem

```
CREATE TABLE student (jméno CHAR (9), prijmeni CHAR (20),
dat.nar DATE, idcislo NUMBER(6), stip NUMBER(6));
```

jméno	prijmeni	dat.nar	idcislo	stip

Tabulka č.1 Základní tabulka, zdroj:vlastní

- **INSERT INTO**

-tento příkaz naplní tabulku

```
INSERT  INTO student
2 VALUES (‘Karel’,’Dvořák’,’19.1.1987’, 111111, 0);
INSERT  INTO student
2 VALUES (‘Tomáš’,’Novák’,’29.10.1989’, 222222, 0);
INSERT  INTO student
2 VALUES (‘Martina’,’Nová’,’4.5.1990’, 333333, 0);
```

-pro zobrazení celé tabulky slouží příkaz **`SELECT * FROM student`**

jméno	prijmeni	dat.nar	idcislo	stip
Karel	Dvořák	19.1.1987	111111	0
Tomáš	Novák	29.10.1989	222222	0
Martina	Nová	4.5.1990	333333	0

Tabulka č.2 Tabulka s daty, zdroj:vlastní

- **ALTER TABLE**

-změna v tabulce, např. přidání sloupce

```
ALTER TABLE student ADD ročník NUMBER (1);
```

- **UPDATE**

-oprava dat v tabulce student

```
UPDATE student SET stip=2000 WHERE idcislo=111111;

UPDATE student SET stip=1500 WHERE idcislo=222222;

UPDATE student SET stip=1000 WHERE idcislo=333333;

UPDATE student SET rocnik=3 WHERE idcislo= 111111;

UPDATE student SET rocnik=3 WHERE idcislo= 222222;

UPDATE student SET rocnik=1 WHERE idcislo= 333333;

SELECT * FROM student;
```

jmeno	prijmeni	Datnar	idcislo	Stip	rocnik
Karel	Dvořák	19.1.1987	111111	2000	3
Tomáš	Novák	29.10.1989	222222	1500	3
Martina	Nová	4.5.1990	333333	1000	1

Tabulka č.3 Tabulka s přidaným sloupcem, zdroj:vlastní

SYNTAXE PŘÍKAZU SELECT:

-příkaz SELECT je základem dotazovacího jazyka SQL

-select = vybrat

- `SELECT * FROM student WHERE stip > 2000;`

  -jelikož nikdo ze studentů nemá stipendium větší jak 2000, nevypíší se žádná data

- `SELECT DISTINCT rocnik FROM student;`

  -tento příkaz slouží k odstranění opakujících se dat (redundantní data)

rocnik
3
1

Tabulka č.4 Odstranění redundantních dat, zdroj:vlastní

- `SELECT * FROM student WHERE stip 2 BETWEEN 1500 AND 2000;`

-vypíší se studenti, kteří mají stipendium ve výši mezi 1500 a 2000

jmeno	prijmeni	Datnar	idcislo	Stip	rocnik
Karel	Dvořák	19.1.1987	111111	2000	3
Tomáš	Novák	29.10.1989	222222	1500	3

Tabulka č.5 Stipendium between, zdroj:vlastní

- `SELECT prijmeni FROM student 2 WHERE prijmeni LIKE 'N%';`

-zobrazí se studenti s příjmením začínajícím na písmeno N

prijmeni
Novák
Nová

Tabulka č.6 Studenti začínající na písmeno N, zdroj:vlastní

- `SELECT * FROM student WHERE rocnik!=3;`

-(!=) nerovnost, tudíž studenti, kteří **ne**chodí do třetího ročníku

jmeno	prijmeni	Datnar	idcislo	stip	rocnik
Martina	Nová	4.5.1990	333333	1000	1

Tabulka č.7 Studenti nechodící do 3.ročníku, zdroj:vlastní

- `SELECT * FROM student WHERE (sysdate-datnar)>9100;`

-výběr na základě hodnoty datumu

jmeno	prijmeni	Datnar	idcislo	stip	rocnik
Karel	Dvořák	19.1.1987	111111	2000	3

Tabulka č.8 Výběr dle data narození, zdroj:vlastní

- `SELECT COUNT (*) AS ”Počet studentů třetího ročníku”`
  `2 FROM student`
  `3 WHERE rocnik=3;`

-zjištění četnosti studentů v 3. ročníku

Počet studentů třetího ročníku
2

Tabulka č.9 Studenti 3.ročníku, zdroj:vlastní

- `SELECT jmeno,prijmeni FROM student`
  `2 GROUP BY rocnik ;`

-tato klauzole seskupí studenty ročníků

jmeno	Prijmeni	rocnik
Karel	Dvořák	3
Tomáš	Novák	3
Martina	Nová	1

Tabulka č.10 Seskupení studentů, zdroj:vlastní

- `SELECT * FROM student ORDER BY datnar ASC;`

-tento příkaz SELECT vybere všechny osoby a setřídí je vzestupně podle data narození

jmeno	prijmeni	Datnar	idcislo	stip	rocnik
Martina	Nová	4.5.1990	333333	1000	1
Tomáš	Novák	29.10.1989	222222	1500	3
Karel	Dvořák	19.1.1987	111111	2000	3

Tabulka č.11 Studenti seřazeni dle data narození, zdroj:vlastní

❖ `SELECT MAX(stip) AS “Nejvyšší stipendium”`
`2 FROM student;`

-zjištění nejvyššího stipendia, příkaz AS umožnuje jakkoli pojmenovat sloupec

Nejvyšší stipendium
2000

Tabulka č.12 Nejvyšší stipendium, zdroj:vlastní

❖ `SELECT jmeno, prijmeni, stip FROM student`
`2 WHERE stip=(SELECT min(stip) FROM student);`

-vypíše se jméno studenta a jeho stipendium, který má nejnižší stipendium

jmeno	Prijmeni	stip
Karel	Dvořák	2000

Tabulka č.13 Student s nejnižším stipendiem, zdroj vlastní

SYNTAXE POHLEDU:

❖ `CREATE VIEW czustudenti`
`2 (jmeno_stud, narozeni, cid, stipendium) AS`
`3 SELECT prijmeni, datnar, idcislo, stip`
`4 FROM student;`
`Pohled vytvořen`

`SELECT * FROM czustudenti;`

jmeno_stud	narozeni	cid	stipendium
Dvořák	19.1.1987	111111	2000
Novák	29.10.1989	222222	1500
Nová	4.5.1990	333333	1000

Tabulka č.14 Pohled, zdroj:vlastní

-Jakmile dojde ke změnám v tabulce student, všechny změny se take projeví v pohledu czustudenti. Pohledy mohou vznikat z jedné tabulky (jednoduché) nebo z více tabulek

(komplexní). Pomocí pohledů je zajištěna bezpečnost dat, uživateli se zpřístupní jen určitá část z tabulky a tím se zabrání poškození nebo zneužití citlivých dat.

- `DROP VIEW czustudenti;`

-pohled byl zrušen

PRÁCE S VÍCE TABULKAMI:

-pro názornou ukázku mějme opět příklad, a to s těmito 3 tabulkami

jmeno	cind	rocnik
Jan Zapletal	12345	3
Jitka Richterová	12344	2

tabulka studenti

cind	predmet	znamka
12345	dat3	2
12344	uč2	1

tabulka prospech

predmet	nazev	katedra
dat3	Databázové systémy 3	KII
uč2	Účetnictví 2	KE

tabulka predmety

- ```
  SELECT jmeno, studenti.cind, prospech.predmet, znamka,
  katedra
  2 FROM student, prospech, predmety
  3 WHERE student.cind=prospech.cind
  4 AND prospech.predmet=predmety.predmet
  5 ORDER BY znamka ASC;
  ```
  ```

-tři tabulky, v každé jsou jiné informace. Pomocí tohoto příkazu jsou tabulky propojeny a vypíší se vybrané položky v vzestupném pořadí dle známky studenta

jmeno	cind	predmet	znamka	katedra
Jitka Richterová	12344	uč2	1	KE
Jan Zapletal	12345	dat3	2	KII

Tabulka č.15 Propojení tabulek, zdroj:vlastní

3.6. Datová integrita

Integrita databáze znamená, že data v ní jsou konzistentní. Data jsou uložena podle předem stanovených pravidel a kritérií. Jedná se o nástroje, které zabraňují k ukládání nesprávných, redundantních či nebezpečných dat, které by mohly poškodit či zneužít tuto databázi.

Druhy Integritních Omezení: Entitní integrita

Doménová integrita

Referenční integrita

3.6.1. Entitní integrita

Toto zabezpečení zajišťuje, aby každý řádek databáze byl jednoznačně rozlišitelný. Řádek (záznam), ve kterém je hodnata PRIMARY KEY musí být jedinečný a nesmí mít hodnotu NULL. Například u rodného čísla či evidenčního čísla.

- CREATE TABLE student
  2 (rodcislo CHAR (11) NOT NULL PRIMARY KEY,
  3 jmeno CHAR (15),
  4 rocnik NUMBER (2));

-pokud se při vkládání jednotlivých záznamů objeví stejná hodnota rodného čísla či rodné číslo nebude vyplněno, bude nahlášena chyba o nesplnění podmínky jedinečnosti.

3.6.2. Doménová integrita

Zajišťuje, že každý atribut (sloupec) bude mít vyžadovaný datový typ a rozsah.

- CREATE TABLE zamestnanci
  2 (evcislo CHAR (4) NOT NULL PRIMARY KEY,
  3 jmeno CHAR (12),
  4 plat NUMBER (5) CHECK (plat BETWEEN 10000 AND 15000));

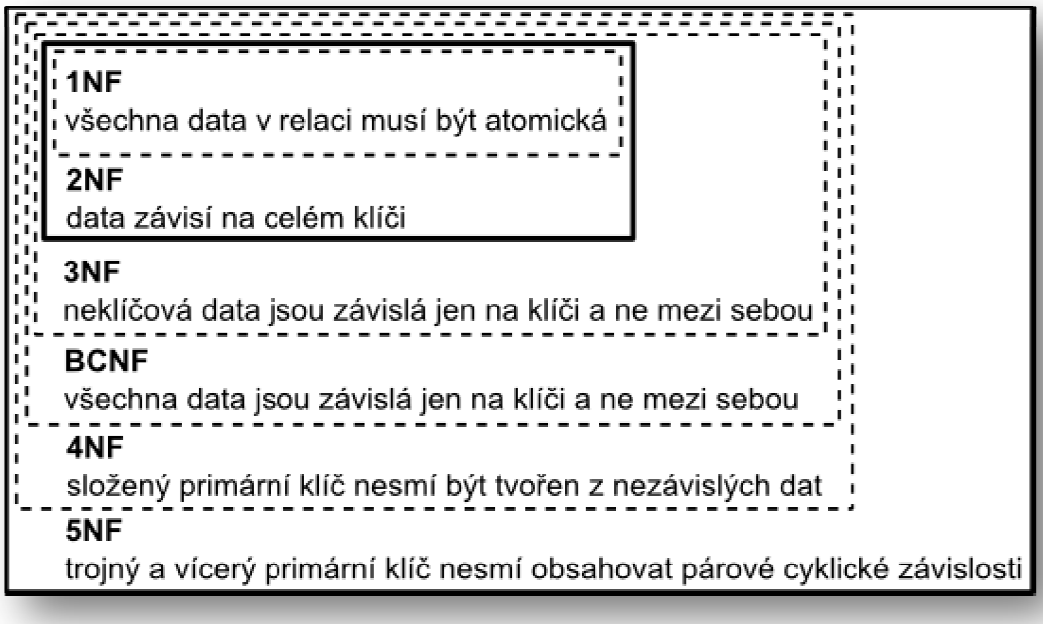
-opět je zde primární klíč, datové typy a jejich velikost, dále pak je stanovena výše platu, pokud tyto podmínky nejsou splněny je hlášena chyba o porušení těchto pravidel

3.6.3. Referenční integrita

Referenční integrita garantuje správnost vztahů mezi jednotlivými tabulkami. Vazby mezi logickými tabulkami nadřízenými a podřízenými a jejich záznamy jsou vytvářeny pomocí primárního klíče (PRIMARY KEY) a cizího klíče (FOREIGN KEY).[5]

3.7. Normalizace

Sada pravidel, která umožnují tvořit vyhovující datové struktury. Relace, neboli tabulky, se rozkládají, aby se s nimi mohlo lépe pracovat, lépe manipulovat a zabraňovat tak rendundanci dat (opakující se data). Tím se sníží nároky na velikost úložného prostoru na disku. Pro databázový stroj je pak práce obecně rychlejší, nemusí zbytečně číst a zapisovat velké množství dat. Pravidla, která by měla data v relaci dodržovat se nazývají Normální formy (NF).



Obrázek č.4 Normální formy, zdroj: http://programujte.com/clanek/2008071900-normalizace-relacnich-databazi/

4. Praktická část

4.1. Popis projektu

V této části bude řešena problematika DB informačního zajištění zájmových sportovních aktivit. Tato záležitost bude demonstrována na problematice basketbalu. Česká basketbalová federace je rozdělena na mužskou část a ženskou část, které dohromady tvoří několik celorepublikových soutěží a krajských soutěží. Pro tuto práci byla vybrána oblast Západní Čechy, přebor basketbalu žen.

Jedná se o soutěž oblastního přeboru žen plzeňského a karlovarského kraje (kvůli nedostatku družstev jsou tyto dva kraje spojeny pod názven Západní Čechy).

Nejdůležitějším aspektem databáze je evidovat družstva a jejich hráčky, které v průběhu sezony získávájí výsledky (statistiky) z utkáních. Tyto výsledky říkají o hráčce jak se chová v utkání, jak je úspěšná, a to kolik nastřílela bodů, kolik doskočila míčů, jak moc byla faulována nebo jak faulovala ona sama a v neposlední řadě kolik odehrála minut za zápas. Z pohledu trenéra jsou to data, díky kterým může ovlivnit svojí taktiku při zápasech. Z pohledu sportovního nadšence jsou to data informační, vyhledává informace jak o družstvech tak o hráčkách, kdy se hraje jaký zápas atd.

Také je důležité evidovat jednotlivá vzájemná utkání se získanými body, které ovlivňují průběh celé soutěže. Dále pak by tato databáze měla obsahovat detaily družstev a detaily jednotlivých hráček jako například: adresa družstva či kontakt na povolanou osobu. To vše tvoří databázi, která je jednou z mnoha složek České basketbalové federace.

4.2. Současný stav a popis problému

Současný stav zadávání a vyhodnocování sportovních událostí nespočívá pouze v naprogramování databáze, ano je to nedílnou součástí, ale není to vše. Budeme-li mluvit o basketbalu vždy je předem určeno místo konání, čas a soupeři. Toto by basketbalová databáze měla již obsahovat, bez těchto informací se dané utkání nemůže konat. Data, která databáze předem neobsahuje jsou bližší informace o utkání (konečné skóré, statistiky). V čase, kdy se utkání odehrává, se zároveň zapisuje „zápis“, který je psán na předtištěný formulář. Někdo by si mohl říct proč se tyto údaje nezadávají přímo do systému? Ano na papír se psalo i ve středověku, má to ale své výhody oproti zadávání do

počítače. Jak člověk tak i stroj není neomylný. Hlavní výhodou je rychlé reagování na dění při hře, přehlednost, obě družstva obdrží okamžitě po skončení utkání kopii zápisu, u psané formy se nemůže stát , že by se systém zhroutil. V nejvyšší soutěži jako je extraliga se už data zadávají přímo do systému, ale stále je přítomen psaný zápis, tento zápis nikdy nevymizí. Dále se při utkání zapisuje tzv. „technický zápis“. Tento zápis není povinný na rozdíl od výše zmiňovaného, ale nezbytný k tomu, aby trenér odhalil věškeré nedostatky svého družstva. Do tohoto zápisu se píší bližší, detailnější informace o dané hráčce. Doskoky útočné a obrané, fauly, trestné hody, body, atd.

Jak probíhá samotné zadávání dat do databáze? Ve vyšších soutěžích jsou na to určení lidé, kteří při utkání zadávají data, starají se o správu a jsou za to i placeni, ale v nižších soutěžích jako je krajský přebor je tomu jinak. Každý z trenérů má přístupové heslo a tím pádem má i nastavená přístupová práva do systému. Po skončení utkání, trenér nebo organizační pracovník, který je pořadatelem, zadává výsledky do databáze. V systému je okamžitě vidět výsledek, který je později zkontrolován správcem. K tomu aby mohl výsledek zkontrolovat mu pomáhá právě zmiňovaný zápis, který je poslán pořadatelem poštou. Posílá se originál ne kopie.

Podobně to funguje i se zadáváním hráček do příslušné databáze. Každá hráčka vlastní licenci. Tento originál je poslán příslušnému orgánu České basketbalové federace. ID lincence (hráčky) je zaevidována do databáze a hráčka je přiřazena do družstva.

Základním problémem těchto databází je jejich neúplnost. Chybí statistiky, základní údaje o hráčkách. V této části bude databáze zpracována, doplněna o chybějící části a bude ukázáno jak by mohla vypadat v prostředí Microsoft Access.

Dalším problémem je nedostatek povolaných osob, které se starají o správu této oblastní soutěže, to však není základním bodem, který je zde řešen. Proto zde bude řešen pouze problém týkající se zajištění sportovních aktivit databázovým systémem.

4.3. Microsoft Access

Aplikace Access je nástroj pro práci s relačními databázemi. Je součástí sady Microsoft office. Vhodný pro osobní potřebu jednotlivých lidí ale i pro menší či větší podniky. Tato aplikace umožnuje rychlou a snadnou práci s informacemi, vytváření sestav díky vylepšenému rozhraní a interaktivních možností návrhu.

Obrázek č.5 Ikona, zdroj: http://www.ctr.cz/microsoft_office/access.htm

Access nevyžaduje důkladnou znalost databází, a to díky předdefinovaným databázovým řešením, které dále můžeme měnit podle potřeb. A i definování těchto potřeb je ulehčeno průvodci.[6]

4.3.1. Historie a vývoj Microsoft Access

V listopadu 1992 byla vydána první verze Microsoft Access 1.0, vzápětí byla vydána verze 1.1 v květnu 1993, která zlepšovala snášenlivost s ostatními produkty Microsoftu a přidávala programovací jazyk Access Basic. Ve stejném roku, tedy 1993, byla vydána verze 2.0, která vyžadovala Microsoft Windows 3.0. Byl vydáván na sedmi 1,44 MB disketách. Tato verze relativně dobře pracovala s malými databázemi avšak později se ukázalo, že za určitých okolností způsobují poškození dat. Největší problémy byly se soubory většími než 10 MB. Spolehlivost databází Access se zvýšila s příchodem Windows 95, 98 a ME. Zvýšila se síťová spolehlivost, postupně se vypustilo 8 opravných balíčků pro Jet Database Engine a zvýšil se také počet uživatelů.

Verze Access 95, verze 7.0, už byla součástí balíku Microsoft Office Profesional spolu s dalšími programy (Excel, PowerPoint, Word). Další změnou bylo nahrazení jazyka Access Basic jazykem Visual Basic for Applications (VBA). S vývojem balíků Office se vyvíjely i verze Accessu.[7]

4.3.2. Základní typy objektů v Microsoft Access

Microsoft Access obsahuje několik typů objektů, které mohou být řešeny v jednom návrhu databáze. Pracuje se s nimi samostatně tak, aby dohromady dávaly smysl a tvořily tak celek.

Tabulky – jsou nejdůležitějším objektem. Od nich se vyvíjejí další typy objektů, jsou na nich přímo závislé. Zde se databáze vytváří. Struktura, podoba, množství tabulek a plnění daty.

Dotazy – podle zadaných kritérií pracuje s tabulkami. Zjišťuje údaje z tabulek definováním objektu dotazy.

Formuláře – zjednodušují práci s tabulkami a dotazy. Je to „grafická maska" pro prohlížení a úpravu dat.

Sestavy – jsou objekty, které jsou určeny pro tisk dat přímo z databáze. Sestava je přímo závisla na tabulce nebo na dotazu. Jedná se o graficky zformátovaná data připravena pro tisk.

Pohledy – virtuální (logické) tabulky, které zpřehledňuji práci s daty a s dotazy. Mají velké uplatnění z hlediska ochrany dat.

4.3.3. Využití Microsoft Access pro tuto práci

Databázové systémy jsou v dnešní době hojně využívány a jsou dostupné již dlouho. Bohužel jsou to systémy buď příliš triviální pro tvorbu aplikací, nebo to jsou složité systémy, které se obtížně učí a používají. Program Microsoft Access je nenáročný a má mnoho příznivců, ti co chtějí vytvářet jednoduché databáze nebo ti, kteří chtějí programovat složité databázové aplikace. Microsoft Access je součástí kancelářského balíku. Je tedy snadno dostupný a nenáročný co se nákladů týče. Na rozdíl od jiných programů pro tvorbu databázových aplikací je také nenáročný z hlediska programování v něm. Není potřeba dokonalých znalostí. To vše šetří čas i finance, které by musela firma, osoba či jiný uživatel vynaložit.

Pomocí aplikace Microsoft Access můžeme[8]:

- Uchovávat data (o družstvech, hráčkách, zápasech a další)
- Pracovat s daty (doplňovat, počítat, měnit)
- Zobrazovat data (prohlížení z různých pohledů)
- Sdílet data mezi uživateli

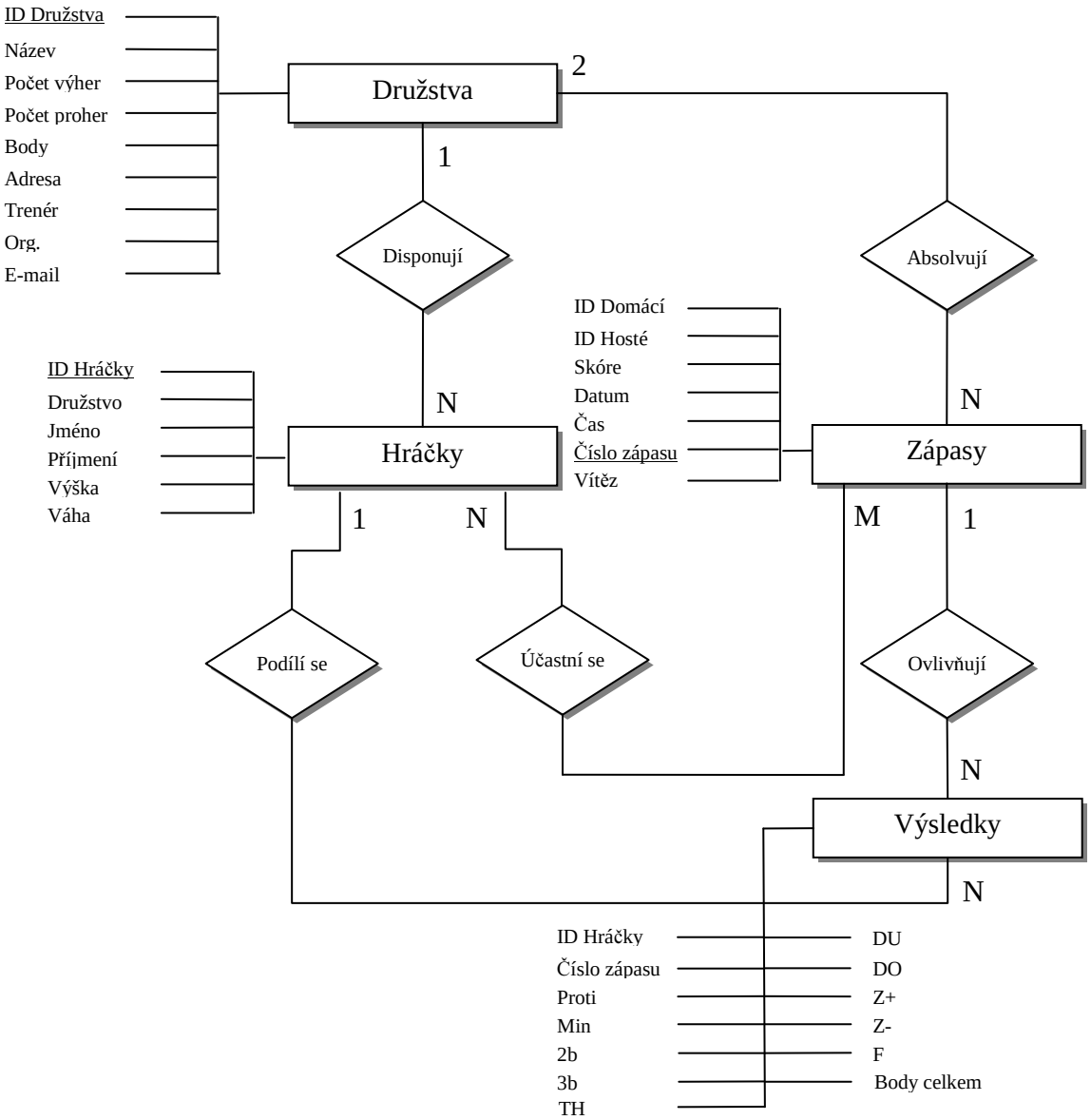
Program Access má i své nevýhody. Databázovou aplikaci lze rozdělit na dvě části. Část datová, ve které se nacházejí tabulky a část aplikační, která nese především dotazy, sestavy a formuláře, tedy část, ve které se aktivně pracuje. Toto rozdělení je ale správné pouze tehdy, kdy je k databázi připojeno 20 a méně uživatelů a velikost databáze je do cca 200 MB. Pokud není databáze rozdělena a je sdílena mezi více uživatelů, není možno v ní dělat úpravy jak dotazů a tabulek tak ani formulářů a sestav. Musí být zajištěno, že žádný uživatel nebude s databází pracovat, aby mohlo dojít ke správě a ke změnám v systému. Do databáze se stále zasahuje, odstraňují se a přidávají se data, a tím dochází ke spomalování výkonu. Proto se musí často dělat komprimace databáze (zmenšení objemu dat), ta se většinou provádí v noci, kdy je předpokládáno, že není uživateli využívána. Tato databáze je tvořena za účelem sdílení dat.

Jak je již na začátku práce zmíněno Česká basketbalová federace je rozdělena na několik celorepublikových soutěží a na soutěže krajské. Budeme-li se zabývat soutěžemi krajskými, tak každý kraj má svého správce nebo skupinu lidí, kteří se o tuto databázi starají a udržují ji v chodu. V tomto případě by bylo možné využít program Microsoft Access, který je nenáročný. Každý kraj by tak spravoval svou vlastní část a nezatěžoval tím tak ostatní. Pokud by to bylo tak, že Česká basketbalová federace by se starala jak o správu krajů tak i o správu celorepublikových soutěží, bylo by výhodnější použít systém, který už je náročnější a finančně odlišný od Microsoftu Accessu.

4.4. E-R model

Nejprve byl navrhnut E-R model pro zjištění struktury databáze a zajištění základních požadavků. Postupovalo se způsobem „shora dolů“, tedy od globálního problému až po jednotlivé kroky:

1. Návrh jednotlivých entit
2. Propojení těchto entit, určení vztahů
3. Doplnění atributů a přiřazení primárních klíčů
4. Určení datový typů jednotlivých atributů



Obrázek č.6 E-R model Oblastního přeboru žen KV 2012/2013, zdroj:vlastní

4.5. Vlastní návrh databáze sportovních aktivit (basketbal) pomocí Microsoft Access 2007

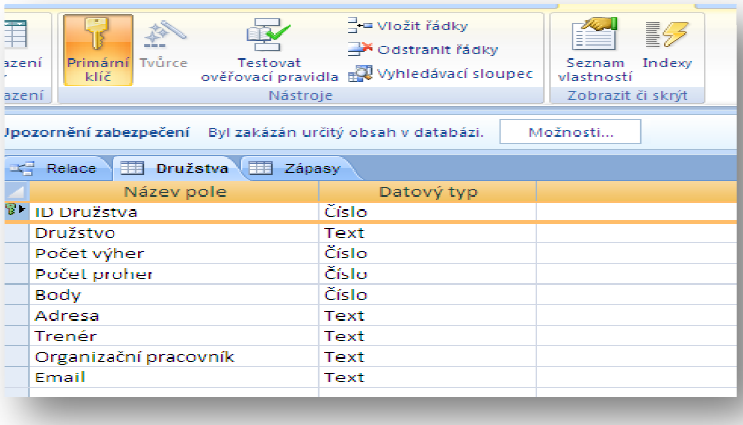
Na začátku je třeba nadefinovat všechny tabulky.Vložení jednotlivých atributů s určitým datovým typem a primární klíč. Základní tabulka: *Družstva*, primární klíč nese atribut ID družstva s datovým typem číslo, datový typ číslo jsou také atributy Počet výher a Počet proher, Body, ostatní mají datový typ text

ID Družstva	Družstvo	Počet výher	Počet proher	Body	Adresa	Trenér	Organizační pracovník	Email
111	BaK Plzeň "B"	6	2	14	Jablonského, Plzeň	Kamila Menoušková	Irena Piksová	bak@seznam.cz
112	BK Klatovy	2	6	10	Voříškova, Klatovy	Dušan Hrabík	Ondřej Sýkora	tjklatovy@seznam.cz
113	BK Sokolov	6	1	13	Jednoty, Sokolov	Lukáš Šnobl	Veronika Šnoblová	sok@email.cz
114	BSK Tatran Kraslice	5	3	13	ČSA, Kraslice	Pavlína Poledňáková	Eva Břízová	basketK@seznam.cz
115	DBK BECKER Kožlany-Kralovice	0	8	8	Jižní, Kožlany	Vladimír Přibyl	Jiří Buňka	kozlany@seznam.cz
116	Slavoj Tachov	5	3	13	Pobřežní, Tachov	Ladislava Pitrová	Miloslav Kohút	basket.tachov@email.cz
117	Sokol Toužim	3	4	10	Školní, Toužim	Romana Pohanková	Petr Gašica	sokoltouzim@seznam.cz

Obrázek č.7 Družstva (ke dni 28.1.2013), zdroj:vlastní

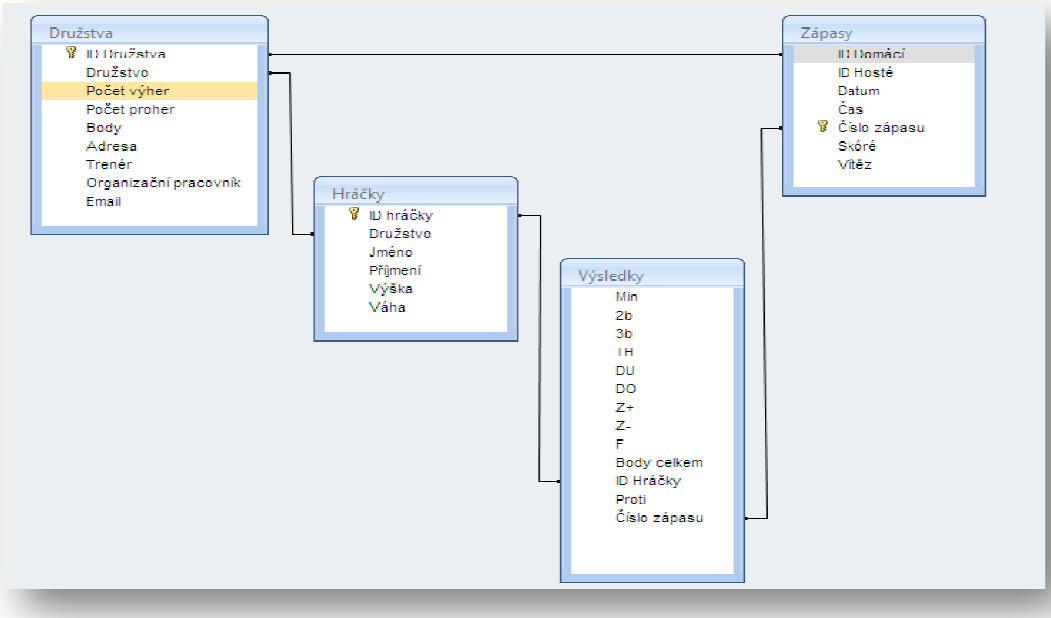
Tabulka: *Zápasy*, primární klíč nese atribut Číslo zápasu, sloupec ID Domácí, ID hosté a číslo zápasu jsou datového typu číslo, atribut čas a datum jak už název vypovídá jsou datovým typem Datum/Čas. Tabulka: *Výsledky*, zde je většina atributů číslem. Tabulka: *Hráčky*, primární klíč je u ID hráčky, který je datového typu číslo stejně tak jako atribut výška a váha, ostatní mají datový typ text.

Doposud byla tabulka v zobrazení datového listu, další možnost zobrazení je návrhové zobrazení, kde se definuje primární klíč či se mění datový typ atributu.



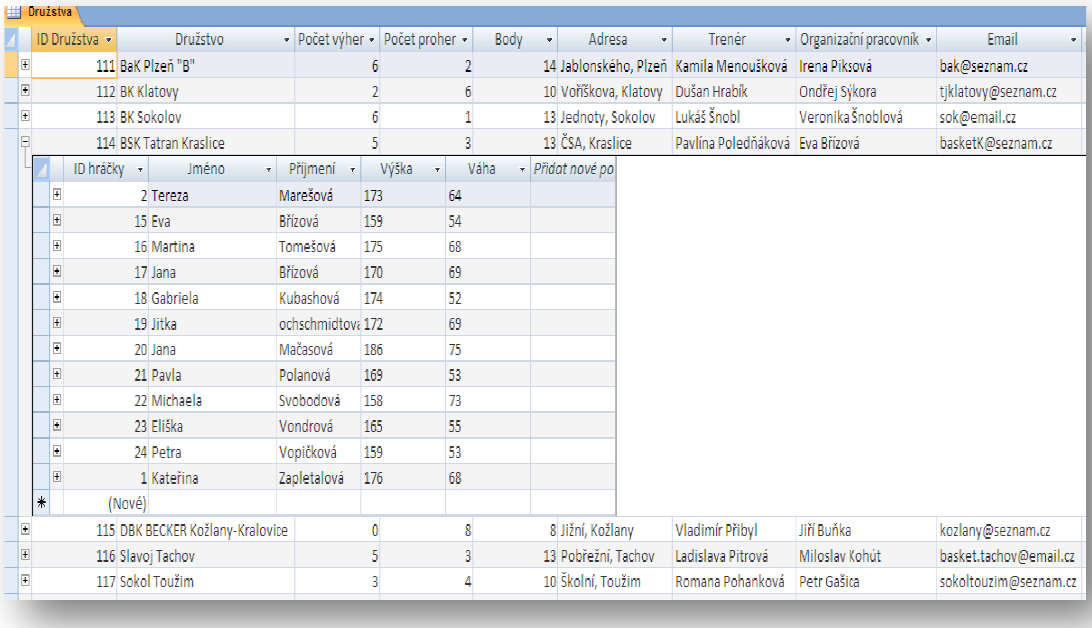
Obrázek č.8 Návrhové zobrazení, zdroj:vlastní

Dalším základním úkonem by mělo být sestavení relace tabulek. Tento úkon se provede pod záložkou databázové nástroje -> vztahy. Zobrazí se nám návrh, se kterým dále manipulujeme. Spojením atributů jednotlivých tabulek docílíme propojení tabulek.



Obrázek č.9 Relace, zdroj:vlastní

Po propojení a vložení primárních klíčů Microsoft Access usnadní orientaci, a to tím, že v tabulce *Družstva* vnoří datový list tabulky *Hráčky*. Pod každým družstvem budou zařazeny jednotlivé hráčky, které tam patří.



Obrázek č.10 Vnořený datový list, zdroj:vlastní

4.5.1. Dotazy

4.5.1.1. Hráčky družstva

Pokud chceme zobrazit hráčky družstva pomocí dotazu, jak je výše zmíněno, není třeba dokonalých znalostí SQL jazyka. Access nás provede průvodcem dotazu, kde je možno si vybrat typ dotazu, jednoduchý dotaz, křížový, vyhledávací na duplicitní řádky nebo vyhledávací na chybějící záznamy. Pro tento případ postačí dotaz jednoduchý.
V dalším kroku budou vybrány tabulky a následně atributy, které budou figurovat ve zvoleném dotazu. Po dokončení dotazu lze jej ještě dále upravovat. Access také automaticky vygenereju SQL příkaz, do kterého také lze zasahovat a dále ho měnit. SQL příkaz, který vypíše hráčky družstva BSK Tatran Kraslice má následující podobu:

```
SELECT Družstva.Družstvo, Hráčky.Jméno, Hráčky.Příjmení,
Hráčky.Výška, Hráčky.Váha

FROM Družstva INNER JOIN Hráčky ON Družstva.Družstvo =
Hráčky.Družstvo

WHERE (((Družstva.Družstvo)='BSK Tatran Kraslice'));
```

4.5.1.2. Zápasy vybraného družstva

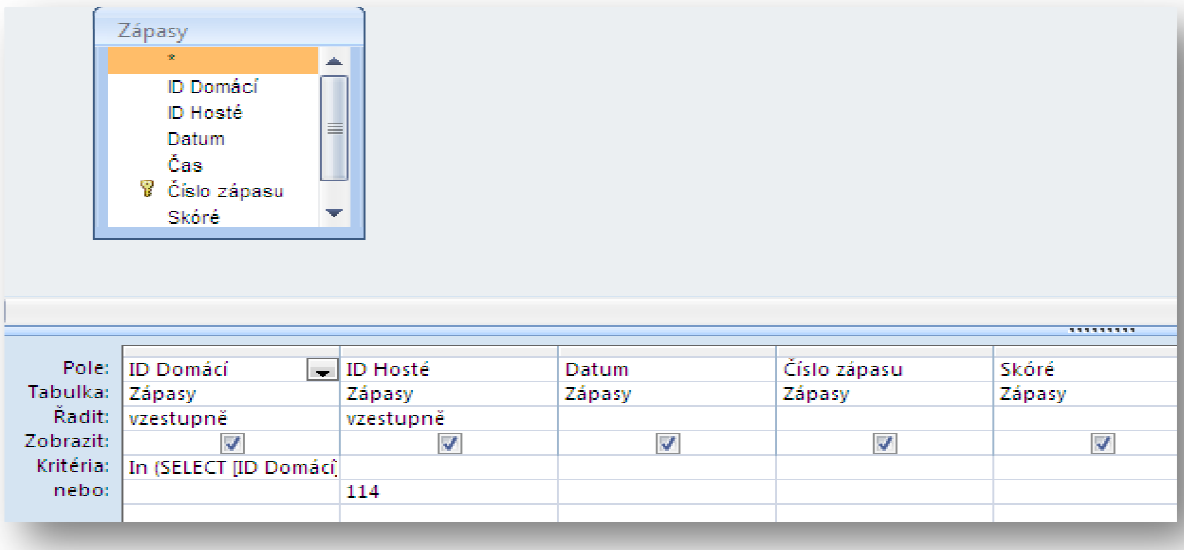
Dalším dotazem v této databázi je dotaz *Zápasy vybraného družstva*. Tabulka zápasů obsahuje všechny zápasy, každý uživatel je příznivcem jiného družstva a proto by chtěl zobrazit zápasy pouze svého favorita. Opět je možno si pomoci průvodcem dotazu. Pro tento případ byl vybrán typ dotazu vyhledávací na duplicitní řádky, tabulka a její atributy k sestavení tohoto dotazu. Nejdříve byly označeny pole, které by mohly obsahovat duplicitní údaje, poté pole, které dále chceme aby tento dotaz obsahoval. Po dokončení se zobrazí návrhové řešení. Stačí už jen zadat parametr, tedy ID Družstva např. 114, které chceme aby se vypsalo. SQL příkaz:

```
SELECT Zápasy.[ID Domácí], Zápasy.[ID Hosté], Zápasy.Datum,
Zápasy.[Číslo zápasu], Zápasy.Skóré

FROM Zápasy

WHERE (((Zápasy.[ID Domácí]) In (SELECT [ID Domácí] FROM
[Zápasy] As Tmp GROUP BY [ID Domácí],[ID Hosté] HAVING
Count(*)>1 And [ID Hosté] = [Zápasy].[ID Hosté]) Or
(Zápasy.[ID Domácí])=114)) OR (((Zápasy.[ID Hosté])=114))

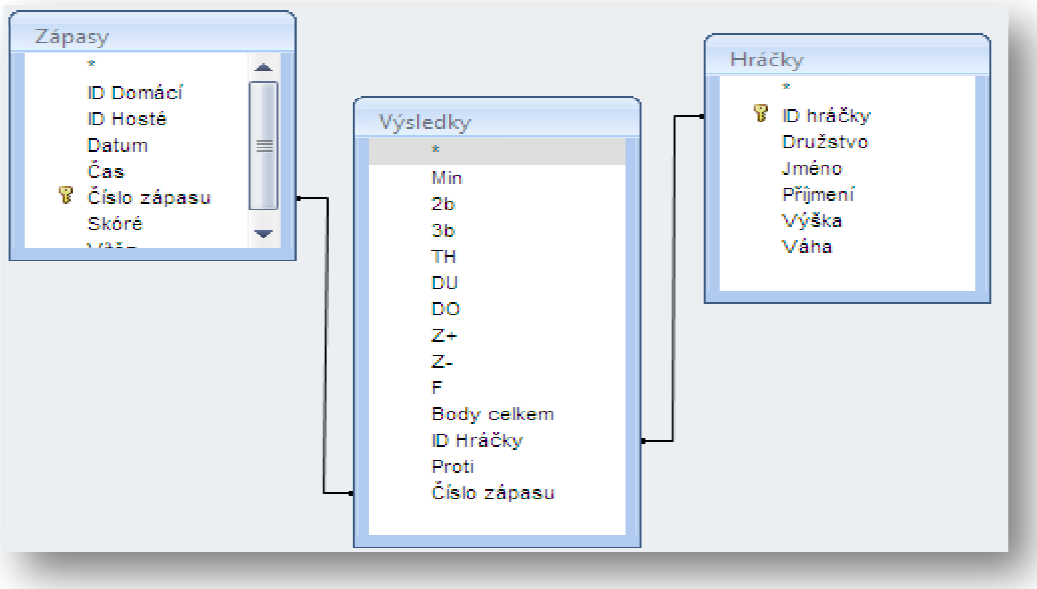
ORDER BY Zápasy.[ID Domácí], Zápasy.[ID Hosté];
```



Obrázek č.11 Zápasy daného družstva, zdroj:vlastní

4.5.1.3. Výsledky hráčky

Neméně důležitým prvkem je dotaz *Výsledky hráčky*. Zde jsou propojeny tři tabulky.



Obrázek č.12 Výsledky hráčky, zdroj:vlastní

Postupuje se stějně jako u předchozích dotazů. Do kolonky kritéria ID Hráčky zadáme námi vybranou hráčku. SQL příkaz bude vypadat takto:

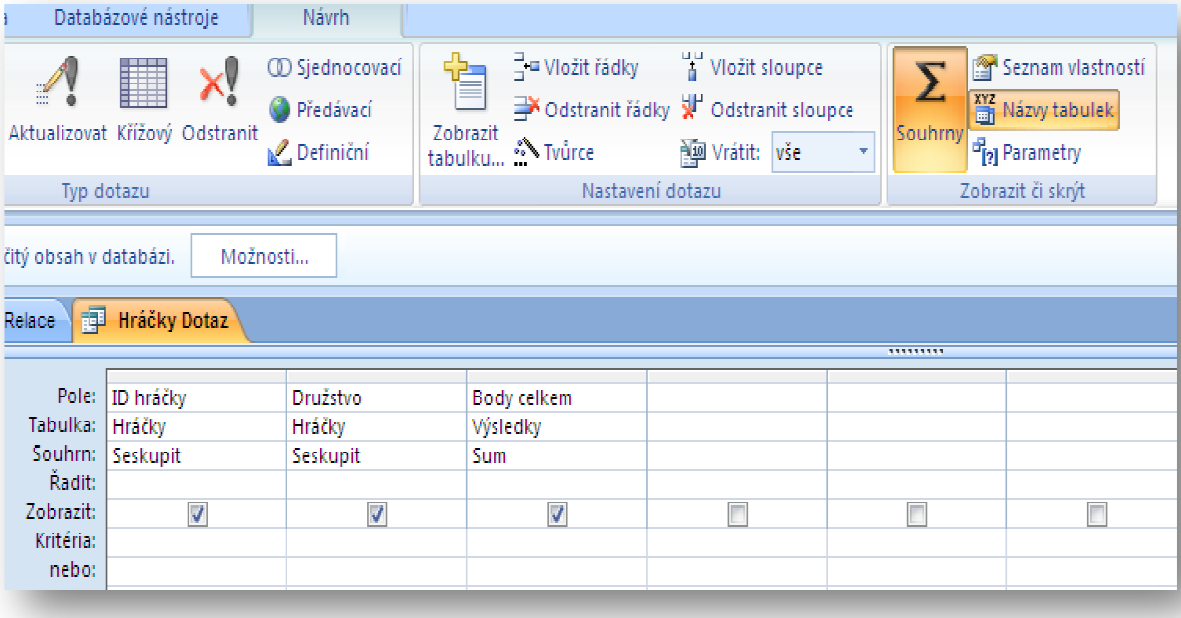
```
SELECT Hráčky.[ID hráčky], Hráčky.Družstvo, Zápasy.[Číslo
zápasu], Výsledky.Proti, Výsledky.Min, Výsledky.[2b],
Výsledky.[3b], Výsledky.TH, Výsledky.DU, Výsledky.DO,
Výsledky.[Z+], Výsledky.[Z-], Výsledky.F, Výsledky.[Body
celkem]

FROM Zápasy INNER JOIN (Hráčky INNER JOIN Výsledky ON
Hráčky.[ID hráčky] = Výsledky.[ID Hráčky]) ON Zápasy.[Číslo
zápasu] = Výsledky.[Číslo zápasu]

WHERE (((Hráčky.[ID hráčky])=7));
```

4.5.1.4. Nejlepší střelec

Nejdůležitějším dotazem pro hráčky samotné je *Nejlepší střelec*. U tohoto dotazu hledáme hráčku, která nastřílela nejvíce bodů. Tzn. seskupit řádky, sestupně, jednotlivých hráček v tabulce *Výsledky* a u každé sečíst body. Vzejde tabulka *Nejlepší střelec*, kde budou hráčky ze všech družstev seřazeny podle počtu nastřílených bodů. V prostředí Accessu si můžeme pomoci průvodcem dotazu, který dále upravujeme v návrhovém zobrazení. V hlavní panelu nástrojů tohoto návrhu označíme položku Souhrny a níže v návrhovém zobrazení zvolíme pod polem Body funkci Sum, která nám sečte body u jednotlivých hráček



Obrázek č.13 Nejlepší střelec, zdroj:vlastní

SQL příkaz pro tento dotaz:

```
SELECT Hráčky.[ID hráčky], Hráčky.Družstvo,
Sum(Výsledky.[Body celkem]) AS [SumOfBody celkem]

FROM Hráčky INNER JOIN Výsledky ON Hráčky.[ID hráčky] =
Výsledky.[ID Hráčky]

GROUP BY Hráčky.[ID hráčky], Hráčky.Družstvo, DESC;
```

4.5.1.5. Družstvo s nejvíce výhrami

U dotazu *Družstvo s nejvíce výhrami* byla použita funkce MAX pro vyhledání družstva, které má nejvíce výher. V tomto případě to byly družstva dvě, obě s 6 výhrami. Funkce MAX a MIN jsou si velmi podobné, pokud bychom použili MIN, vypsána budou družstva s nejmenším počtem výher. SQL příkaz s vnořeným SELECTEM:

```
SELECT Družstva.[ID Družstva], Družstva.Název,
Družstva.[Počet výher], Družstva.Trenér

FROM Družstva

WHERE (((Družstva.[Počet výher])=(SELECT Max(Družstva.[Počet
výher]) FROM Družstva)));
```

4.5.1.6. Pořadí družstev

Nejdůležitějším dotazem této databáze je *Pořadí družstev*. Z tabulky *Družstva* se vypíší všechny atributy. U atributů počet výher a body bylo použito klíčové slovo DESC, které řadí zvolené sloupce vzestupně. Toto by mělo zajistit, že na prvním místě bude družstvo s nejvíce body a nejvíce výhrami. Musí se také brát ohled na to, kolik družstvo odehrálo utkání, snadným dopočítání se zjistí, že například družstvo Plzně má 8 odehraných zápasů a družstvo Toužimi jen 7. Kdyby i tak došlo ke shodě, řídí se rozhodování vzájemnými zápasy a jejich skóré, tedy dopočítáním vzájemného skoré. Např. první výsledek mezi družstvem A a B(A-domácí, B-hosté) je 50:57, druhý výsledek B proti A je 50:58(B-domácí, A-hosté), v konečném součtu A:B 108:107, vyhrává družstvo A. SQL příkaz:

```
SELECT Družstva.[ID Družstva], Družstva.Název,
Družstva.[Počet výher], Družstva.[Počet proher],
Družstva.Body

FROM Družstva

ORDER BY Družstva.[Počet výher] DESC , Družstva.[Počet
proher], Družstva.Body DESC;
```

ID Družstva	Název	Počet výher	Počet proher	Body
111	BaK Plzeň "B"	6	2	14
113	BK Sokolov	6	1	13
116	Slavoj Tachov	5	3	13
114	BSK Tatran Kraslice	5	3	13
117	Sokol Toužim	3	4	10
112	BK Klatovy	2	6	10
115	DBK BECKER Kožlany-Kralovice	0	8	8

Obrázek č.14 Celkové pořadí, zdroj: vlastní

Toto byly základní dotazy pro databázi. Ostatní by fungovaly na stejném či podobném principu jako zde jmenované. Nejpoužívanějším dotazem by tedy byl dotaz SELECT.

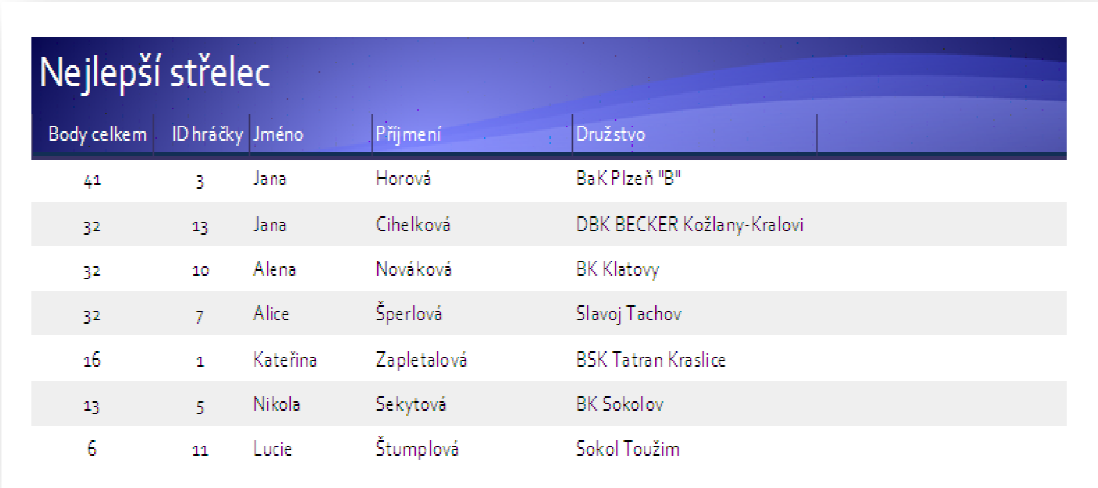
4.5.2. Sestavy

Sestavy jsou graficky zformátovaná data určená pro tisk. Jsou přímo závislá na tabulkách či dotazech. Nejzásadnější sestava je *Pořadí družstev* ze stejnojmeného dotazu. Opět Access usnadňuje práci díky průvodci sestavou. Zadávájí se atributy, které mají figurovat v sestavě a styl, grafika, návrhu. Také je možno návrh upravovat z hlediska designu, barvy písma, okrajů, odstavců atd. Následují sestavy, které znázorňují další možné grafické úpravy.

Pořadí družstev

ID Družstva	Název	Počet výher	Počet proher	Body
111	BaK Plzeň "B"	6	2	14
113	BK Sokolov	6	1	13
116	Slavoj Tachov	5	3	13
114	BSK Tatran Kraslice	5	3	13
117	Sokol Toužim	3	4	10
112	BK Klatovy	2	6	10
115	DBK BECKER Kožlany-Kralovice	0	8	8

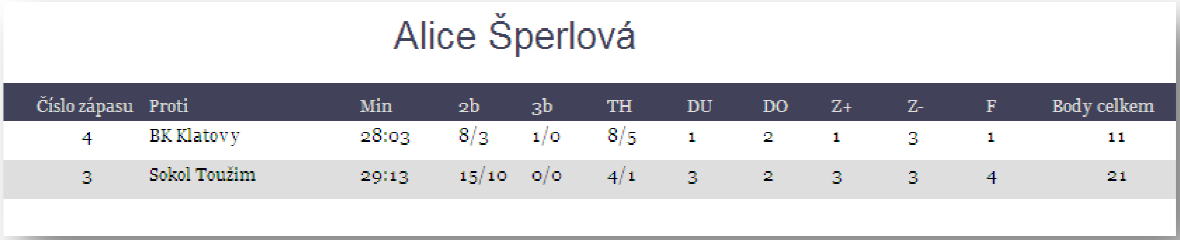
Obrázek č.15 Sestava pořadí družstev, zdroj: vlastní



Nejlepší střelec

Body celkem	ID hráčky	Jméno	Příjmení	Družstvo
41	3	Jana	Horová	BaK Plzeň "B"
32	13	Jana	Cihelková	DBK BECKER Kožlany-Kralovi
32	10	Alena	Nováková	BK Klatovy
32	7	Alice	Šperlová	Slavoj Tachov
16	1	Kateřina	Zapletalová	BSK Tatran Kraslice
13	5	Nikola	Sekytová	BK Sokolov
6	11	Lucie	Štumplová	Sokol Toužim

Obrázek č.16 Sestava nejlepší střelec, zdroj: vlastní



Alice Šperlová

Číslo zápasu	Proti	Min	2b	3b	TH	DU	DO	Z+	Z-	F	Body celkem
4	BK Klatovy	28:03	8/3	1/0	8/5	1	2	1	3	1	11
3	Sokol Toužim	29:13	15/10	0/0	4/1	3	2	3	3	4	21

Obrázek č.17 Sestava statistika hráčky, zdroj: vlastní

4.5.3. Interpretace výsledků

Z této databáze lze vyčíst mnoho výsledků, které jsou důležitými informacemi jak pro členy týmu tak pro uživatele zvenčí. Struktura družstev, pořadí družstev se získanými body, vzájemné zápasy s konečnými skóré, statistiky hráček a nejrůznější sestavy či žebříčky. Stále ale lze přidávat funkce, které by mohly napomoct ke zlepšení výkonu družstva či hráčky samotné. Další funkcí by mohla být hodnota, která vypočítá jak moc je hráčka aktivní, užitečná, důležitá pro družstvo. Tzv. koeficient užitečnosti. Do tabulky *Družstva* se vložil další atribut datového typu číslo. Toto číslo udává jak moc těžký je tento soupeř pro ostatní, tedy čím výše družstvo figuruje v tabulce pořadí tím vyšší číslo bude mít.

Pořadí	Název	Koeficient
1.	Bak Plzeň „B“	3,5
2.	BK Sokolov	3
3.	Slavoj Tachov	2,5
4.	BSK Tatran Kraslice	2
5.	Sokol Toužim	1,5
6.	BK Klatovy	1
7.	DBK BECKER Kožlany-Kralovice	0,5

Tabulka č.16 Vložený koeficient, zdroj: vlastní

V tabulce *Výsledky* se také přidal další atribut datového typu číslo. V tomto sloupci se počítá užitečnost dané hráčky pro každý zápas, pro každý záznam hráčky. Pro tento případ je mnoho možností jak by se dal koeficient počítat, zde byl vybrán způsob následující:

Příklad: Hráčka z družstva na 4. pořadí absolvovala zápas proti družstvu, které je na první příčce tabulky a má koeficient 3,5.

[(Body celkem + DO + DU + Z+)*3,5]-[(Z- + F)*3,5]= Číslo

Vyhodnocení: Získaná čísla by se porovnávala mezi sebou, zákonitě čím vyšší číslo bude tím lépe, tím výšší užitečnost bude u dané hráčky.

SQL příkaz:

```
SELECT ((Body Celkem+DO+DU+(Z+))*3,5)-(((Z-)+F)*3,5) AS
‘Užitečnost’

FROM Výsledky

WHERE ID Hráčky=7 AND Číslo zápasu=4;
```
[9]

Užitečnost
38,2

Tabulka č.17 Užitečnost, zdroj: vlastní

5. Závěr

Bakalářská práce byla tématicky zaměřena na db systémy v informačním zajištění sportovních akcí. V práci byly vymezeny teoretické principy relačních databází, popsán jazyk SQL a jeho syntaxe, datová normalizace a integritní omezení. V praktické části byla použita aplikace Microsoft Access ke zpracování výše zmíněného tématu. Zdrojem informací a zároveň podkladem pro tuto práci byla existující databáze basketbalu, a to konkrétně soutěže Oblastní přebor žen v basketbalu, Západní Čechy, které se autorka aktivně zúčastňuje. Byly použity existující jména, nikoliv soukromé údaje. Soukromé údaje byly nahrazeny údaji smyšlenými. Celá praktická část je věnována právě této oblastní soutěži. Všechny řešené a zmíněné problémy se týkají pouze této soutěže.

Cílem práce bylo zmapovat současnou úroveň využití db technologie v informačním zajištění sportovních akcí a identifikovat bariéry jejího širšího uplatnění a navrhnout odstranění těchto bariér.

Současná evidence sportovních aktivit, se během několika let měnila. V počátcích se většina dat zaznamenávala na papír, tyto data se dále musela ověřovat a přeposílávat, což bylo zdlouhavé. Například zařazování hráče/hráčky a jeho licence (hráčské karty) do družstva. Licence se musela poslat na schválení České basketbalové federaci (ČBF), poté putovala zpět a až tehdy mohl hráč nastoupit. Tento proces byl urychlen, údaje o licenci se zadají do příslušné databáze, schválí a hráč může nastoupit do utkání. Těmito ale i dalšími změnami přispěly relační databázové systémy k vylepšení fungování evidence sportovních aktivit.

Nedostatkem již existující databáze je její neúplnost (chybějící údaje, záznamy, funkce a statistiky). Tyto nedostatky vedou ke zkresleným či nedostatečným informacím. Mohou také vést k tomu, že družstvo není schopno se zlepšit, nevidí své nedostatky nebo nedostatky druhých. To, že družstvo disponuje zkušenostmi a dovednostmi nezaručuje vždy nejlepší výsledky. Právě díky dostatku informací, které družstvo může získat z dobře fungující evidence, je možné dosahovat lepších výsledků.

Pomocí aplikace Microsoft Access autorka vypracovala databázi basketbalu výše zmíněné soutěže.

Přínosnost:

- detailnější analýza dat o družstvu (kladné či záporné vlastnosti)
- nové informace, které trenér využívá při taktice nebo při trénincích
- zefektivnění výkonu družstva
- větší informovanost veřejnosti
- základní vymezení principů práce s Microsoft Access

Závěrem lze konstatovat, že byla prokázána možnost vytvoření relačně databázové aplikace, která by byla využitelná. Díky nástroji Microsoft Access by byla práce pro uživatele ekonomicky přijatelná a nenáročná. Dále by pak byla i snádná distribuce mezi ostatní uživatele. Těžko lze na dnešním trhu najít srovnatelný nástroj na tvorbu relačních databází, který by byl finančně přijatelný a zároveň by nevyžadoval programovací schopnosti.

6. Seznam obrázků

7. Seznam tabulek

8. Použitá literatura

[1] FairCom. *SELECT*FROM SQL History* [online].[cit. 2012-10] dostupné z:

< http://www.faircom.com/ace/enl_22_s12_t.php>

[2] *Databáze* [online]. 2010 [cit. 2012-10] Dostupné z:

< http://www.databaze.chytrak.cz/modely.htm>

[3] *Database Directory* [online].[cit. 2012-10] Dostupné z:

< http://www.databasedir.com/db2/>

[4] SHELDON, R. *SQL začínáme programovat*. Praha, Granda Publishing, 2005. 1.vydání. ISBN 80-247-0999-6

[**5**] VOSTROVSKÝ, V. *Vytváření databází v Oracle*. Praha, Česká zemědělská univerzita v Praze, 2009. 1.vydání. ISBN 9788021311916

[6] MICROSOFT. *Access 2007* [online]. 2013 [cit. 2013-01] Dostupné z:

<http://www.ctr.cz/microsoft_office/access.htm>

[7] ČEČÁK, Ondřej. *Microsoft Office Access*[online]. 2009[cit. 2013-2] Dostupné z: <http://www.cecak.cz/fel/dba/referaty/access/historie_vlastnosti_pouziti_microsoft_office_access?do=show>

[8] KRUCZEK, A. *Microsoft Office Access 2007*. Brno, Computer press, 2007. 1.vydání ISBN 978-80-251-1608-1

[9] LACKO, L. *1001 tipů a triků pro SQL*. Praha, Computer press, 2011 ISBN 9788025130100