

# Obsah

<b>Předmluva</b> .....	<b>3</b>
<b>1. Úvod</b> .....	<b>5</b>
1.1 Databáze .....	5
1.2 Vývoj databázových systémů.....	6
1.3 Společnost Oracle.....	7
<b>2. Databázový systém Oracle8i</b> .....	<b>8</b>
2.1 Úvodní popis .....	8
2.1.1 Objektově relační přístup .....	8
2.1.2 Klient / Sever .....	8
2.1.3 Správa databáze .....	9
2.1.4 Podpora Javy .....	10
2.2 Základní komponenty Oracle8i .....	10
2.2.1 Oracle8i server .....	10
2.2.2 Oracle8i klient.....	12
2.2.3 Enterprise Manager.....	12
<b>3. Příprava dat</b> .....	<b>15</b>
3.1 Návrh datové základny .....	15
3.2 Vytvoření databáze .....	16
3.3 Vytvoření objektů v databázi.....	18
3.3.1 Vstup do databáze .....	18
3.3.2 Tabulkový prostor a datový soubor .....	19
3.3.3 Nový uživatel resp. schéma.....	20
3.3.4 Vytvoření tabulek.....	21
3.2 Vstup dat.....	23
<b>4. Možnosti prezentace a správy dat na Internetu</b> .....	<b>25</b>
4.1 Úvod .....	25
4.2 Web Publishing Assistant .....	25
4.2.1 Princip .....	25
4.2.2 Vytvoření webové stránky .....	26
4.2.3 Šablony .....	27
4.2.4 Konkrétní příklad .....	28
4.3 Oracle WebDB .....	30

4.3.1 Úvodní popis .....	30
4.3.2 Struktura a funkce WebDB .....	31
4.3.3 Práce s WebDB .....	34
4.3.4 Vytvoření ukázkové aplikace .....	37
4.4 ODBC / JDBC .....	40
<b>5. Návrh a vytvoření webové databázové aplikace .....</b>	<b>42</b>
5.1 Volba software .....	42
5.1.1 PHP .....	43
5.1.2 MySQL .....	43
5.1.3 Další nástroje .....	43
5.2 Návrh datové základny .....	44
5.3 Úvodní informace .....	46
5.3.1 Příprava v databázi .....	46
5.3.2 Hlavní funkce PHP pro komunikaci s databází přes ODBC .....	46
5.3.3 Poznámky ke stavbě skriptů .....	47
5.4 Struktura a funkce aplikace .....	48
5.5 Budoucí vývoj aplikace .....	51
<b>6. Závěr .....</b>	<b>53</b>
<b>7. Dodatky .....</b>	<b>55</b>
A. Instalace Oracle8i Enterprise Edition 8.1.6 .....	55
B. Instalace Oracle WebDB 2.2 .....	56
C. Instalace a konfigurace Apache, PHP a MySQL .....	57
D. Konfigurace ODBC .....	59
<b>8. Zdroje informací .....</b>	<b>61</b>
<b>9. Přílohy .....</b>	<b>63</b>
9.1 Šablona – Oracle Web Publishing Assistant .....	63
9.2 Definice kaskádových stylů CSS .....	63
9.3 Zdrojové kódy skriptů vytvořené databázové aplikace .....	65

## Předmluva

Internet... V posledních deseti letech nejdynamičtější fenomén informačních technologií vstoupil téměř do všech oblastí lidské činnosti. Je to dost možná jen logický důsledek toho, že jednou z nejdůležitějších komodit se staly informace. Internet je totiž především prostředek, jak informace sdílet. Vezmeme-li v úvahu rychlost nárůstu počtu uživatelů, přenosové rychlosti a průchodnosti telekomunikačních sítí a vývoje technologií, není třeba pochybovat o světlé budoucnosti (alespoň co se informačních technologií týče).

Stejně jako mnoho uživatelů, i můj kontakt s „informační dálnicí“ začínal u prohlížení World Wide Webu a elektronické pošty, jakožto rychlým a v praktickém smyslu neosobním prostředkem komunikace. Dalším krokem byla touha přijít se svou „troškou do mlýna“ v podobě vlastních WWW stránek. Zřejmě v této oblasti vývoj nejviditelněji postoupil (a postupuje). Od pouhé kombinace textu s grafickými prvky přešel k dynamicky generovaným stránkám, které jsou v převážné většině založené na datech z databází.

Jednou z největších výhod Internetu je, že umožňuje pomocí standardního webového prohlížeče přistupovat k databázi prakticky komukoli, má-li na to oprávnění. Propojení www stránky s databázovými systémy je jeden z nejdůležitějších interaktivních prvků dnešních webových aplikací. Umožňuje shromažďovat údaje, přistupovat k nim, číst informace z databáze a publikovat je prostřednictvím World Wide Webu.

Právě zájem o zmíněné technologie byl základním impulsem pro výběr (a samozřejmě i vypracování) této diplomové práce – *Databázový systém Oracle v prostředí sítě Internet*.

V úvodní části jsou stručně vysvětleny nejdůležitější pojmy z teorie a historie relačních databází a krátké seznámení se společností Oracle. Následující dvě kapitoly jsou věnovány charakteristice základních vlastností, principů a součástí databázového systému Oracle8i a dále praktickému popisu práce s hlavními nástroji pro práci s objekty v databázi.

V pořadí čtvrtá kapitola se zabývá dostupnými nástroji Oracle8i pro prezentaci a správu dat na Internetu – jedná se o Oracle Web Publishing Assistant (kapitola 4.2) a samostatnou komponentu Oracle WebDB (kapitola 4.3). U obou nástrojů jsou

uvedeny jejich hlavní funkce a způsob práce s nimi, včetně praktického předvedení na datech pro tento účel připravených v kapitole 3. Dále je stručně vysvětlen princip univerzálního rozhraní pro přístup k databázím (kapitola 4.4), které je mimo jiné použito v závěrečné (praktické) části mé práce.

Ve zmíněné poslední kapitole je popsán návrh a vytvoření vlastní databázové aplikace pro správu a prezentaci výsledků úloh přes webové rozhraní. Aplikace je napsána ve skriptovacím jazyce PHP ve spojení s databází MySQL.

Hlavní zaměření této práce spočívá v charakteristice možností prezentace a správy dat na Internetu. Neposkytuje ucelený popis relačních databází,

ani se nezabývá všemi vlastnostmi systému Oracle8i. Jsou vysvětleny jen hlavní pojmy použité v souvislosti se zmíněnými webovými aplikacemi (totéž platí o dotazovacím jazyku SQL, PHP a HTML).

Pro vznik této práce byly použity programy a nástroje určené pro platformu Windows (NT resp. 9x). Stejný (nebo podobný) software je k dispozici i pro Linux (včetně volné verze databázového systému Oracle8i a WebDB), avšak mé zkušenosti s tímto operačním systémem nejsou, jak se ukázalo, natolik hluboké, abych dokázal řešit četné problémy (nejen) s instalací databáze Oracle. Díky společnosti Oracle Czech s.r.o. jsem měl pro účely této práce zapůjčen databázový systém Oracle8i a Oracle WebDB 2.2 pro Windows NT.

Instalace a konfigurace hlavních použitých programů je popsána v dodatkových kapitolách a v následujících přílohách jsou uvedeny výpisy nově vytvořených PHP skriptů a důležitých souborů.

# 1. Úvod

## 1.1 Databáze

Databáze je místo, kde jsou uloženy logicky uspořádané informace (data). Tímto místem nemáme již dnes na mysli skříň se zásuvkami, ale nějaké záznamové médium počítače. Ve své podstatě není mezi skříní a záznamovým médiem velký rozdíl. Počítač nám však umožňuje pracovat s daty efektivněji – tzn. zpracovávat rychleji větší objemy dat.

Data a nástroje pro jejich správu tvoří *databázový systém* [3]:

***Databázový systém = Data + Nástroje pro práci s daty***

Nástroje pro práci s daty zajišťují například tyto činnosti (různé databázové systémy se od sebe liší mimo jiné vybaveností těmito nástroji):

- vytvoření, vyhledání, aktualizaci a rušení dat
- definici struktury dat
- definici a zajištění integrity dat (soulad dat s reálným stavem)
- zajištění fyzické<sup>1</sup> a logické<sup>2</sup> nezávislosti dat
- podpora práce více uživatelů
- zálohování dat

Většina dnešních databázových systémů používá pro uspořádání údajů v databázi *relační model dat*. V tomto modelu jsou data uspořádána do tabulek s přiděleným jménem, které ji v rámci databáze identifikuje. Tabulka obvykle obsahuje informace o jednom druhu objektu. Skládá se z určitého počtu sloupců (minimálně jednoho), kterým říkáme *atributy* či *položky* a řádků. Každý sloupec má přiděleno jméno a *datový typ* (např. text, číslo, datum, logické hodnoty – viz.více v kapitole 3.1). Jednotlivé řádky se nazývají *záznamy* a tabulka jich může (teoreticky) obsahovat libovolný počet. Od nulového počtu (prázdna tabulka) až po počet omezený kapacitou sloupce resp. tabulky (záleží na daném databázovém systému). Každá hodnota v tabulce je tak jednoznačně identifikována jako průsečík sloupce a řádku. Sloupec je dán svým jménem a řádek *identifikátorem řádku*. Úlohu tohoto identifikátoru plní v tabulce tzv. *primární klíč*. Je to položka (sloupec), jehož hodnota

---

<sup>1</sup> Fyzická nezávislost = oddělení způsobu fyzického uložení dat (např. na disku) od způsobu práce s nimi

<sup>2</sup> Logická nezávislost = změna logické struktury dat nevyžaduje úpravu existujících programů a dotazů

je v rámci jedné tabulky jedinečná. Primární klíč může tvořit i kombinace více položek. Většinou se jedná o číselnou hodnotu, u které máme jistotu, že se nebude měnit (např. rodné číslo osoby nebo identifikační číslo objektu).

Přístup k informacím uloženým v databázi zajišťuje program zvaný *SŘBD (Systém Řízení Báze Dat)* na principu klient-server. Název pochází z anglického DBMS (DataBase Management System). SŘBD je nepřetržitě spuštěn (jako služba na Windows NT resp. démon na Unixu) a v roli serveru očekává požadavky klientů (ostatních aplikací).

Pro zadávání požadavků SŘBD zpravidla slouží jazyk *SQL (Structured Query Language)*. Tento jazyk vznikl v sedmdesátých letech ve firmě IBM pod názvem SEQUEL a poté SE-QUEL/2. V osmdesátých letech byl přejmenován na SQL. Protože ho využívaly i další firmy, vznikla potřeba standardizace. Výsledkem toho byl v roce 1986 standard SQL86 a dále v roce 1992 standard SQL2. V současné době se chystá standard SQL3, který bude odrážet směřování k objektovým databázím (více v [3]). I přes tyto standardy existují mezi jednotlivými databázovými systémy odlišnosti v syntaxi některých příkazů.

## 1.2 Vývoj databázových systémů

První databázové aplikace se začali objevovat v polovině padesátých let v podobě jednoduchých evidencí. V šedesátých letech vytvořila firma IBM první databázový systém založený na hierarchickém modelu (podoba stromové struktury organizace). Relační databáze jsou na světě od sedmdesátých let - jako první opět firma IBM v roce 1977 s databází SYSTEM R a v roce 1979 firma Relational Software, Inc. (dnes *Oracle Corporation*) s databází *Oracle*. IBM dále v roce 1981 uvedla systém SQL/DS a později v roce 1983 DB2. Své systémy produkovaly i další firmy jako např. Progress, Informix<sup>3</sup>, SyBase či Microsoft.

V současné době se vývoj ubírá směrem k objektovým databázím, které umožňují uchovávat široké spektrum dat jako je text, obraz, zvuk a video. Buď jsou to čistě objektově založené databázové systémy nebo relační systémy s implementovaným objektovým přístupem, jejichž výhodou je zpětná kompatibilita s dříve napsanými programy (více v [3]).

---

<sup>3</sup> V roce 2001 se Informix spojil s IBM

První počítačové programy byly vyvinuty k provádění matematických operací, ale současné době (či spíše v posledním desetiletí) stojí na prvním místě databázové aplikace. Ať už jde o jednoduché evidence, registrační systémy nebo programy, které využívají svůj vlastní databázový formát, přestože to není na první pohled zřejmé – jako například CAD systémy. Poměr aplikací, které některou databázovou technologií využívají vůči těm, které tuto podporu nepotřebují se dnes odhaduje na 7:3 [7].

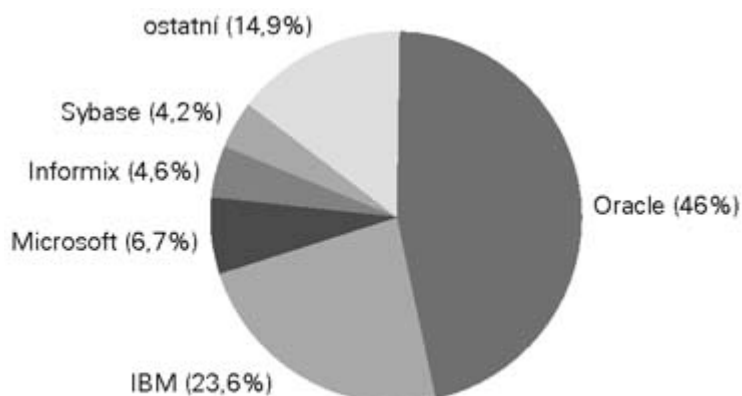
V budoucnu se budou jistě stále prosazovat i v oblastech, kde donedávna nehrály příliš významnou roli – typickým příkladem jsou dynamicky generované webové stránky.

### 1.3 Společnost Oracle

Oracle Corporation (dále jen Oracle) je druhá největší softwarová společnost na světě a zároveň vedoucím dodavatelem technologií pro e-business a webové aplikace. Je první softwarovou společností, která orientovala vývoj celé škály svých produktů výhradně na internet. Společnost Oracle byla

založena v roce 1979 jako Software Development Laboratories, Inc. a v roce 1982 převzala jméno Oracle podle původního názvu vyvíjeného systému správy databází pro sálové počítače. Má sídlo v Redwood Shores v Kalifornii.

Česká pobočka (Oracle Czech s.r.o.) byla založena v roce 1994 a sídlí v Praze ve Škrétově ulici č. 12.



Zdroj: IDC, 2001

Obr. 1.3: Celkový podíl na trhu relačních databázových licencí v roce 2000.

## 2. Databázový systém Oracle8i

### 2.1 Úvodní popis

Společnost Oracle získala významný podíl na trhu hi-end databázových systémů již v roce 1991 uvedením databáze Oracle7. Další vývojový stupeň znamenala databáze Oracle8 a po ní verze Oracle8i, která byla navržena speciálně pro vývoj a provoz aplikací na Internetu. Právě Oracle8i (konkrétně Oracle8i Enterprise Edition 8.1.6 for MS Windows NT) jsem měl k dispozici pro tuto práci.

#### 2.1.1 Objektově relační přístup

Oracle8i je objektově relační SQL databáze typu klient/server. I nadále zůstává plně podporován relační databázový model a navíc jsou k dispozici určité objektové vlastnosti. Z tohoto pohledu se systém nazývá *hybridní*.

Základní objektově orientované vlastnosti:

- *Objektové typy* – umožňují vytvářet uživatelské datové typy a těmi pak v objektových tabulkách nahradit standardní datové typy
- *Objektové pohledy* – slouží v případě, že potřebujeme přistupovat k datům uloženým v relační tabulce pomocí objektových technologií
- *Objektová rozšíření* – definování a práce s objekty pomocí nových příkazů jazyků SQL a PL/SQL<sup>4</sup>

Vzhledem k povaze problematiky a zaměření této práce odkazují na publikaci [1] nebo obecně na zdroje o objektovém programování.

#### 2.1.2 Klient / Sever

Důležitou charakteristikou relačních databázových systémů je skutečnost, že fyzická struktura databáze (soubory na disku, do kterých se ukládají databázové informace) je před koncovým uživatelem skryta. Uživatel vidí pouze logickou strukturu dat (databázové tabulky) a nemusí nic vědět o jejich fyzickém uspořádání. O mapování dat ze sloupců a řádků tabulek do souborů uložených na disku se stará databázový server. Výhodou databází klient/server oproti tradičním PC databázím (např. dBASE,

---

<sup>4</sup> PL/SQL (Procedural Language for Structured Query Language) – Procedurální jazyk strukturovaného dotazovacího jazyka. Jde o množinu příkazů jazyka SQL rozšířenou o procedurální prvky [4].



FoxPro) je kromě rychlosti a spolehlivosti také rozdělení zátěže mezi klientský systém a databázový server. Na straně klienta běží aplikace, pomocí které uživatel pracuje s daty. Databázový server provádí správu datových zdrojů pro více klientů. Teoreticky lze obě části systému spustit na jednom stroji, ale tím zanikne řada výhod tohoto typu aplikací.

### 2.1.3 Správa databáze

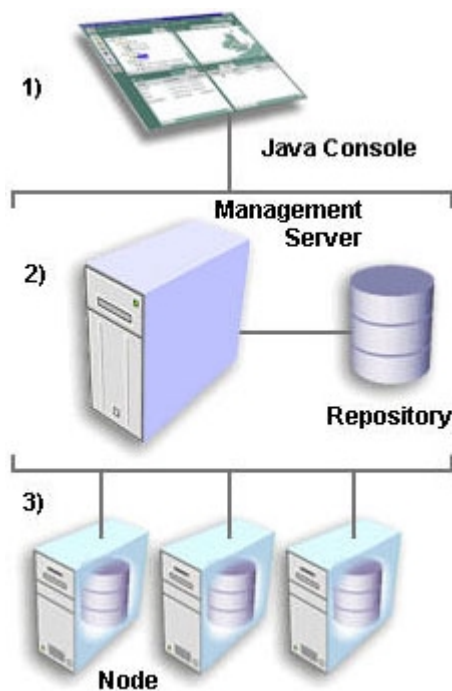
Základní rozhraní pro provádění všech administrátorských úkonů představuje *Oracle Enterprise Manager*. Aplikace využívá třívrstvou architekturu, která umožnila přenést většinu zátěže na server (viz. obr. 2.1.3).

První vrstvu tvoří správcovská konzole (*Enterprise Manager Console*) a integrované aplikace. Grafické uživatelské rozhraní obsahuje menu, nástrojové lišty a 4 konfigurovatelné panely:

- **Navigator** - zobrazení jednotlivých databází a objektů<sup>5</sup> v databázi ve stromové struktuře
- **Group** - umožňuje vytvářet logická seskupení databázových serverů nebo objektů na grafickém pozadí (například na mapě)
- **Jobs** - automatické spouštění úloh (umožňuje plánovat nebo vzdáleně spouštět např. zálohování)
- **Events** - monitorování předdefinovaných událostí (např. dojde-li k důležité události, automaticky se provede příslušná akce)

Z prostředí konzole se lze připojit k libovolné oraclovské databázi v síti a spouštět řadu nástrojů určených pro řízení jednotlivých operací (např. vytváření a rušení objektů, spouštění či ukončení běhu databáze).

Druhou vrstvu tvoří jeden nebo více *Management Serverů*. Jejich úkolem je zpracovávat a řídit distribuci všech zadaných úloh. Pro uchování systémových i



Obr. 2.1.3: Třívrstvá architektura

<sup>5</sup> V tomto případě (a i v dalších kapitolách) jsou pojmem objekty míněny objekty v databázi – např. tabulky, uživatelé, role apod.

aplikačních dat a informací o stavu jednotlivých uzlů (*nodes*) používá *Enterprise Manager Repository*.

Třetí vrstvu představují jednotlivé servery v síti, které prostřednictvím agentů (*Intelligent Agent*) vykonávají příslušné úlohy.

Některé konkrétní úkony v databázi (včetně jejího vytvoření) budou popsány v kapitole 3.

#### **2.1.4 Podpora Javy**

Jedním z charakteristických rysů Oracle8i je široká podpora Javy. Java (přesně řečeno *Java Virtual Machine*) je integrována přímo do databázového stroje. Výsledkem je, kromě zvýšení spolehlivosti a rychlosti, možnost v databázi psát, ukládat a spouštět libovolný Java kód. To přináší mnoho předností – nezávislost na platformě, vytváření inteligentních objektů a opakovatelné využití komponent.

Oracle8i podporuje také jazyk *SQLJ*, který umožňuje použít SQL příkazy v programech napsaných v Javě a komunikovat tak s databází. Na vývoji tohoto standardu se podílely firmy Oracle, IBM a Sun. *SQLJ* lze použít jak na straně klienta, kdy přístup k databázi zajišťuje rozhraní *JDBC*<sup>6</sup> (viz. více v kapitole 4.4), tak na straně serveru, kde Java aplikace pracují s databází přímo.

### **2.2 Základní komponenty Oracle8i**

Jak již bylo řečeno, systém je rozdělen na dvě základní části: na stranu serveru a stranu klienta.

V následujících odstavcích budou stručně představeny jen ty nejzákladnější komponenty systému Oracle8i Enterprise Edition. Některé součásti, které se přímo týkají zaměření této práce, budou detailněji popsány v následujících kapitolách. Podrobnější informace viz. [1] nebo originální dokumentace.

#### **2.2.1 Oracle8i server**

Serverová část systému zahrnuje samotné databázové jádro a procesy běžící na pozadí, které zajišťují chod vlastní databáze. Obsahuje také elementární nástroje, které umožňují takové základní operace jako spouštění a zastavení databáze. Tyto utility i samotná databáze se chovají na všech platformách naprosto stejně – rozdíly

---

<sup>6</sup> *JDBC* – Java DataBase Connectivity. Více viz. kapitola 4.4

jsou ukryty přímo v jádře, kde Oracle využívá jednotlivé platformy k uložení dat, jejich čtení a zápisu.

### **Server Manager**

Nebo též *SVRMGR* - umožňuje provádět důležité operace jako spouštění a zastavování databáze nebo přímo zadávat příkazy jazyka SQL.

### **Net8**

Tento nástroj zajišťuje přenos dat a příkazů z databázového serveru k ostatním komponentám, které se nacházejí vně základních utilit – tzn. spojuje serverovou část s částí klientskou (popř. spojuje server-server) a data uložená v databázi. Z toho plyne, že je součástí serveru i klientů. Net8 zároveň umožňuje síťové připojení z dalších nástrojů, které nejsou standardní součástí základního programového balíku.

### **SQL\*Plus**

Tento nástroj umožňuje vytvářet a spouštět SQL dotazy, vkládat záznamy nebo měnit data. Může být spuštěn na straně serveru i klienta. „Plus“ v názvu značí, že obvyklý rozsah SQL příkazů byl rozšířen mimo jiné o možnost přizpůsobovat výstupní sestavy (např. nastavit velikost stránky, záhlaví a zápatí), upravovat a ukládat soubory, definovat proměnné, výzvy pro vstupy od uživatelů nebo vyvolávat příkazy operačního systému.

### **EXP80 a IMP80**

Pomocí těchto utilit je možno data exportovat (EXP80) z databáze nebo je do ní importovat (IMP80). Takovýmto způsobem lze data z tabulky exportovat do souboru, ten poté např. přenést na jinou platformu (např. na Unix) a naimportovat je do databáze.

### **Loader**

Tento nástroj umožňuje do databáze vložit data z jiné databázové platformy. Data je možno načíst (po nastavení příslušných parametrů) ze standardního textového souboru, který obsahuje tabulky.

Vedle tohoto nástroje již dnes existují další možnosti. Můžeme využít například standardu ODBC (viz. více v kapitole 4.4) nebo asistentů systému Oracle. Ti slouží k propojení např. se systémy MS SQL Server nebo MS Access a většinou je možno stáhnout je zadarmo z webových stránek Oracle<sup>7</sup>.

---

<sup>7</sup> Oficiální webové stránky Oracle Corporation: <http://www.oracle.com>

## **Precompilers**

Předkompilátor je překladač, který umožňuje vývoj aplikací v různých jazycích využívajících vložených příkazů jazyka SQL pro přístup do databáze. K dispozici je standardně jeden předkompilátor pro každý podporovaný programovací jazyk jako COBOL, Ada, C, C++ nebo FORTHRAN (ne na NT). Množství podporovaných jazyků se liší podle platformy.

### **2.2.2 Oracle8i klient**

Klientská část systému se skládá z utilit a pomocných programů, které je možné spustit na libovolném počítači v síti. Jejich součástí je obvykle také grafické uživatelské rozhraní, které velmi usnadňuje ovládání systému.

#### **Net8 a SQL\*Plus**

Tyto komponenty jsou součástí serverové i klientské části systému – jejich stručný popis viz. kapitola 2.2.1

#### **Assistants**

Asistenti jsou pomocníci, kteří vedou uživatele některými úkoly jako například přenos dat z Oracle7 nebo jiných databázových platform (MS Access apod.). Další takovým úkolem je překlacení relačních databázových struktur do objektových. Jeden ze standardních asistentů také napomáhá vytvořit webové stránky – podrobněji viz. kapitola 4.2.

#### **Enterprise Manager**

Enterprise Manager je nejdůležitější komponentou klientské části. Proto bude blíže popsán v samostatné (následující) kapitole a prakticky předveden v kapitole 3.

### **2.2.3 Enterprise Manager**

Tato sada nástrojů slouží k plnění mnoha administrátorských úkonů jako například spravovat databázové objekty nebo řešit problémy výkonosti. Sada umožňuje centrálně spravovat mnoho databází (ať už lokálních či vzdálených). Některé nástroje jsou standardní součástí Oracle8i, jiné se dodávají zvlášť.

Většina nástrojů poskytuje grafické uživatelské rozhraní, které umožňuje vykonávat mnoho administrátorských úkonů bez nutnosti zadávat příkazy jazyka SQL. Máme však možnost nechat si SQL kód zobrazovat ve spodní části okna a tam ho podle potřeby doladovat. Zároveň nám to umožňuje vygenerovaný kód uložit do souboru a

ten poté spustit ve chvíli, kdy chceme akci v opakovat (bez nutnosti znovu procházet všechny ovládací prvky).

### **Enterprise Manager Console**

Tento nástroj byl popsán v kapitole 2.1.3 – Správa databáze.

### **DBA Studio**

Tato komponenta zobrazuje ve stromové struktuře všechny dostupné databáze a objekty v nich. V této struktuře je možno objekty třídit podle typů objektů nebo podle schémat<sup>8</sup> (vlastníků). Zároveň v rámci databáze poskytuje přístup k nejdůležitějším nástrojům pro její správu:

- **Schema Manager** – umožňuje interaktivně vytvářet, mazat a upravovat tabulky, pohledy<sup>9</sup>, synonyma<sup>10</sup>, integritní omezení apod.
- **Security Manager** – s jeho pomocí lze spravovat uživatele, role<sup>11</sup> a profily<sup>12</sup>
- **Instance Manager** – poskytuje přehled všech činností právě probíhající nad jednou databází (*instancí*) - např. aktuálně přihlášení uživatelé nebo jejich činnosti. Zároveň umožňuje databázi spouštět a zastavovat.
- **Storage Manager** – pomocí tohoto nástroje můžeme změnit prostor pro databázi, sledovat a měnit datové soubory a tabulkové prostory<sup>13</sup>.

Z hlediska nutnosti (a možnosti) zadávat příkazy SQL platí skutečnosti uvedené v úvodu této kapitoly (2.2.3).

### **Data Manager**

Tento nástroj umožňuje provádět export a import dat, spouštět skripty nebo zpřístupnit data ke sdílení jiné databázi.

### **Backup Manager a Recovery Manager**

Tyto utility poskytují průvodce pro ukládání dat na disk nebo pásku, různé varianty záloh a v případě nutnosti obnovení databáze ze zálohových souborů.

---

<sup>8</sup> SCHÉMA je dáno uživatelským jménem uživatele – zahrnuje a identifikuje všechny objekty, které daný uživatel v rámci databáze vytvořil.

<sup>9</sup> POHLED je fiktivní tabulka vzniklá vybráním určitých sloupců (popř. i řádků) z původní tabulky. Tímto způsobem je možno zpřístupnit jen určitá data z původní tabulky.

<sup>10</sup> SYNONYMA jsou alternativní pojmenování databázových objektů, které většinou slouží ke zjednodušení jmen. Ty je možno používat v SQL dotazech.

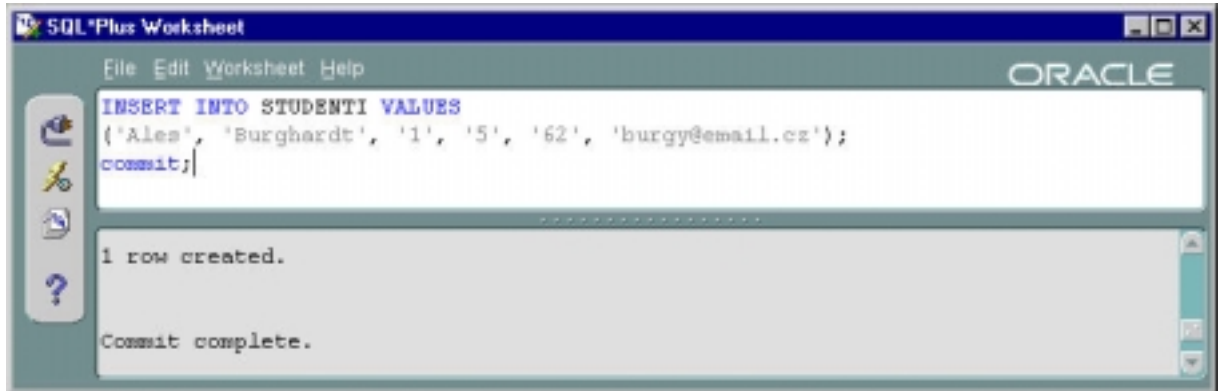
<sup>11</sup> ROLE je skupina práv. Uživatelům se stejnými právy k databázi není nutno tyto práva jednotlivě definovat, ale stačí jim přiřadit příslušnou roli.

<sup>12</sup> PROFIL je, podobně jako role, skupina vlastností, která se jako celek přiřadí uživateli. V tomto případě se jedná o vlastnosti jako omezení systémových zdrojů, doba připojení nebo maximální počet připojení jednoho uživatele. V Oracle8i standardně existuje jeden implicitní profil *Default*.

<sup>13</sup> TABULKOVÝ PROSTOR je logický prostor, kterému je přiřazen jeden nebo více fyzických datových souborů.

## SQL\*Plus Worksheet

Tato utilita má dělené okno – v horní části je možno zadávat SQL (SQL\*Plus, PL/SQL) příkazy a v dolní části se zobrazuje výsledek. Umožňuje editovat příkazy, vracet se zpět v historii a znovu příkazy spouštět. Zároveň dovoluje kód (i výsledek) ukládat do souborů a uložené zpětně spouštět.



Obr. 2.2.3: SQL\*Plus Worksheet

### 3. Příprava dat

V této kapitole bude demonstrováno vytvoření nové databáze, uživatele, tabulkového prostoru apod. Poté budou vytvořeny jednoduché tabulky a naplněny daty. Na těchto datech budou v dalších kapitolách předvedeny možnosti prezentace a správy dat na internetu.

#### 3.1 Návrh datové základny

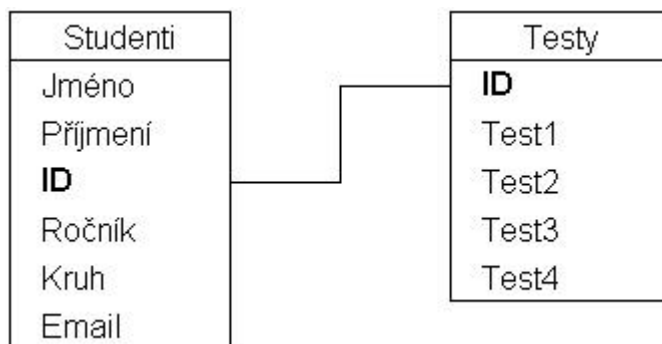
Pro ilustraci byla zvolena fiktivní studijní skupina, u které budou evidovány zapsaní studenti (včetně doprovodných údajů) a jejich výsledky jednotlivých testů v daném předmětu.

Datovou základnu budou tvořit dvě tabulky (viz. Obr. 3.1):

- **Studenti** – Jméno, Příjmení, Identifikační číslo, Ročník, Studijní kruh a Kontaktní email
- **Testy** – Identifikační číslo, Test 1, Test 2, Test 3 a Test 4

Jako unikátní identifikační číslo může sloužit např. rodné číslo – zde pro tento účel poslouží jednoduché pořadové číslo. Toto identifikační číslo bude zvoleno jako *primární klíč* v tabulce *Studenti* a zároveň jako odkaz v tabulce *Testy* – tzv. *cizí klíč*.

Tabulka *Testy* obsahuje výsledky z jednoho konkrétního předmětu. Pro další předměty by existovaly totožné tabulky (s jiným jménem popřípadě jiným počtem testů – sloupců). Jiná, výhodnější, struktura datové základny pro konkrétní aplikaci bude popsána v kapitole 5.



Obr. 3.1: Relační schéma

**Datové typy jednotlivých sloupců (viz. přehled níže):**

- Jméno – VARCHAR2 (15)
- Příjmení – VARCHAR2 (20)

- ID, Kruh – NUMBER (2,0)
- Ročník, Test – NUMBER (1,0)
- Email – VARCHAR2 (40)

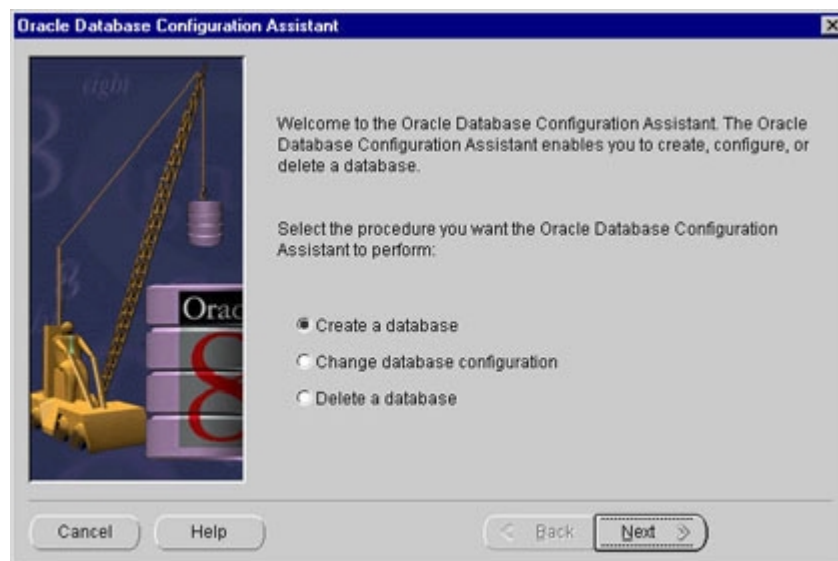
### Nejpoužívanější datové typy Oracle8i [1]:

<i>Datový typ</i>	<i>Parametry</i>	<i>Popis</i>
CHAR(d)	d = 1 až 2000	Řetězec znaků s pevnou délkou. Implicitní délka je 1, maximální délka je <i>d</i> .
DATE		Datum v rozsahu 1.1. 4712 př. n. l. až 31.12. 4712 n. l. Je to sedmibajtové číslo, které obsahuje i čas v hodinách, minutách a sekundách.
FLOAT(p)	p = 1 až 126	Reálné číslo. Parametr <i>p</i> určuje přesnost.
LONG		Řetězec znaků s proměnnou délkou (max. 2 GB).
NCHAR(d)	d = 1 až 2000	Jako typ CHAR + podpora národních abeced.
NUMBER(p,d)	p = 1 až 38 d = -84 až 127	<i>p</i> je počet číslic <i>d</i> je počet desetinných míst
VARCHAR2(d)	d = 1 až 4000	Řetězec znaků proměnné délky, kde <i>d</i> je maximální počet znaků.

### 3.2 Vytvoření databáze

Úvodní databázi může (volitelně) vytvořit již instalátor Oracle8i. Jsou v ní předvytvořeny všechny základní struktury jako uživatelé, role, „defaultní“ profil, tabulkové prostory s datovými soubory apod.

Pokud jsme databázi při instalaci nevytvořili nebo chceme vytvořit novou, použijeme *Database Configuration Assistant*. Tento asistent zároveň umožňuje existující databázi smazat či změnit její parametry.



Obr. 3.2 – 1: Vytvoření databáze pomocí Database Configuration Assistant



## Postup:

1. Po spuštění asistenta vybereme *Create a database* a pokračujeme ⇒ *Next*
2. Zvolíme *Typical* (vytvoří standardní databázi s minimální nutností zásahů ze strany uživatele – pouze identifikace databáze – viz. bod 4) ⇒ *Next*
3. Jako metodu zvolíme *Copy existing database files from the CD* (databáze s běžným nastavením – dostupné soubory jsou nakopírovány z CD – nejrychlejší metoda vytvoření). Pokud chceme parametry volit ručně nebo CD nemáme, zvolíme druhou možnost. ⇒ *Next*

### 4. Identifikace databáze:

**Global Database Name** je plné jméno databáze, které ji identifikuje v rámci celé sítě. Zadává se ve tvaru:

database\_name.domain // jméno\_databáze.jméno\_domény

Jméno domény lze najít v menu *Start* ⇒ *Settings* ⇒ *Control Panel* ⇒ *Network*.

Můj počítač není v síti a doména je pouze pojmenována DOMAIN.

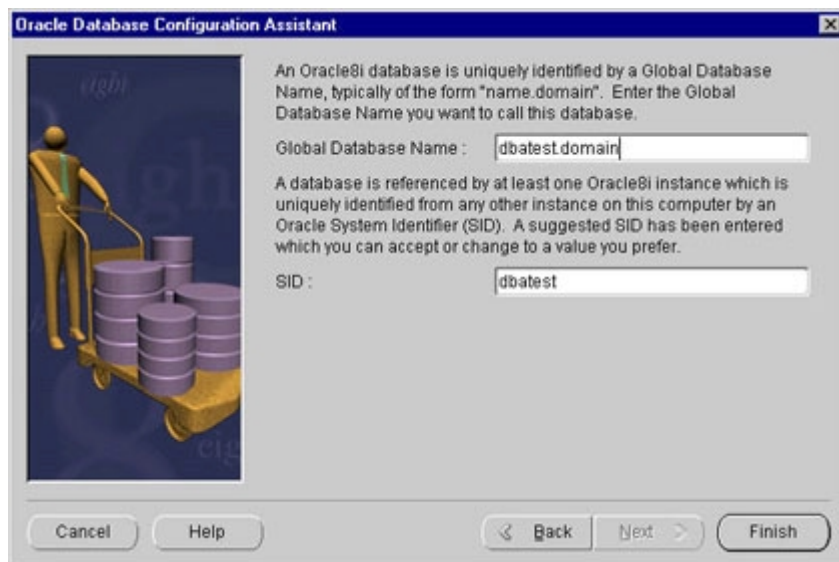
**SID** (*System Identifier Database*) je jméno databáze unikátní v rámci počítače.

Defaultně je to část *database\_name* z *Global Database Name*, ale lze změnit.

Moje konkrétní hodnoty:

**Global Database Name:** DBATEST.DOMAIN

**SID:** DBATEST



Obr. 3.2 – 2: Identifikace nově vytvořené databáze

5. Potvrdíme hlášení o umístění souborů databáze. Implicitně je to *oracle\_base\oradata\database\_name* – zde konkrétně (viz. dodatek A): *D:\Oracle\oradata\test2*

6. Potvrdíme hlášení o jméně databáze a o systémových uživateli s hesly. Hlavní předinstalovaní uživateli nové databáze jsou:

<i><b>Uživatel</b></i>	<i><b>Heslo</b></i>	<i><b>Popis</b></i>
SYSTEM	manager	DBA <sup>14</sup>
SYS	change_on_install	Systémový DBA
INTERNAL	oracle	Alias pro SYS

7. Po několika minutách proběhne vytvoření databáze. Potvrdíme informační hlášení o Global Database Name, SID a přístupových heslech (viz. bod 6).

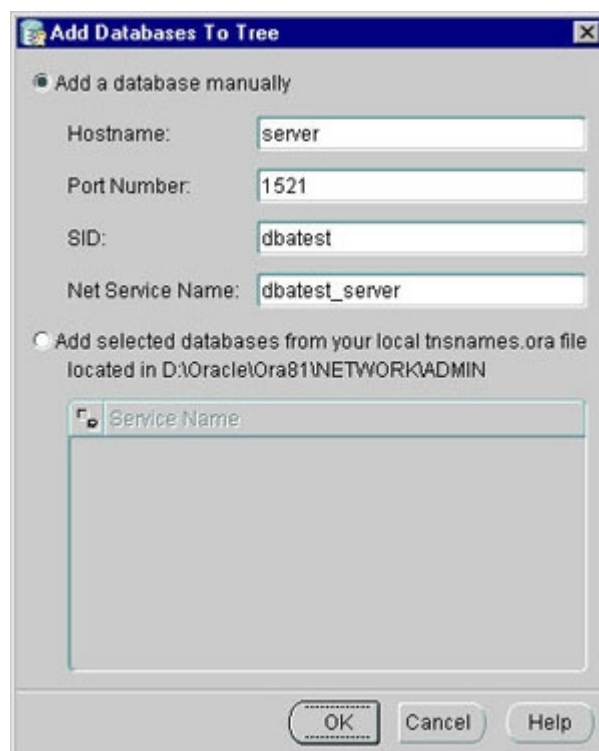
### 3.3 Vytvoření objektů v databázi

Nyní budou v databázi vytvořeny potřebné objekty pomocí nástroje *DBA Studio*, které poskytuje požadované funkce v grafické podobě. Zároveň s tím bude uveden vygenerovaný SQL kód, který by vytvořil stejné objekty např. v prostředí *SQL\*Plus Worksheet*.

#### 3.3.1 Vstup do databáze

Po spuštění *DBA Studio* je nutné nově vytvořenou databázi „přidat do stromu“. To provedeme příkazem z menu *File* ⇒ *Add Database To Tree...* a v následném okně vyplníme hodnoty v poli *Add a database manually*:

- *Hostname*: SERVER (= jméno počítače, viz. *Start* ⇒ *Settings* ⇒ *Control Panel* ⇒ *Network*)
- *Port Number*: 1521 (ponecháme)
- *SID*: DBATEST (= jméno databáze - viz. kapitola 3.2, bod 4)
- *Net Service Name*: DBATEST\_SERVER (generuje se automaticky)



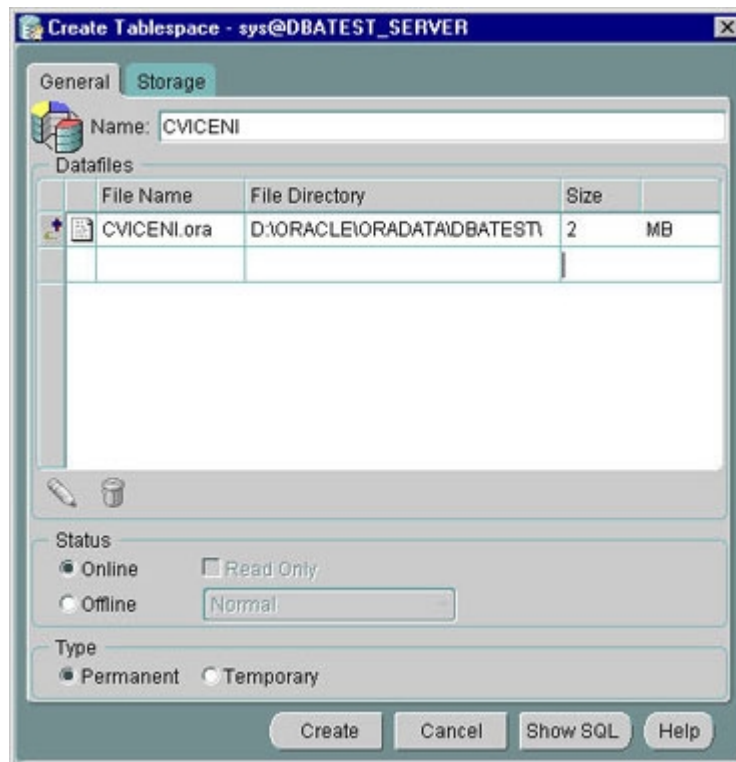
Obr. 3.3.1: Přidání nové databáze do stromu v *DBA Studio*

<sup>14</sup> DBA (DataBase Administrator) – Databázový administrátor

Ve stromu v levé části okna se objeví nová databáze pod jménem *DBATEST\_SERVER* (resp. jméno služby – Service Name). K databázi se připojíme příkazem z menu *File* ⇒ *Connect...* Přihlásíme se jako databázový administrátor (*Username = SYS*), který má implicitní heslo (*Password*) *change\_on\_install*. Po přihlášení se v rámci databáze zobrazí jednotlivé správcovské nástroje, které budou použity v dalších kapitolách.

### 3.3.2 Tabulkový prostor a datový soubor

Tabulkový prostor (a zároveň i datový soubor) vytvoříme pomocí nástroje **Storage Manager**. Z menu zvolíme *Object* ⇒ *Create...* nebo použijeme ikonu v levé nástrojové liště.



Obr. 3.3.2: Vytvoření tabulkového prostoru a datového souboru

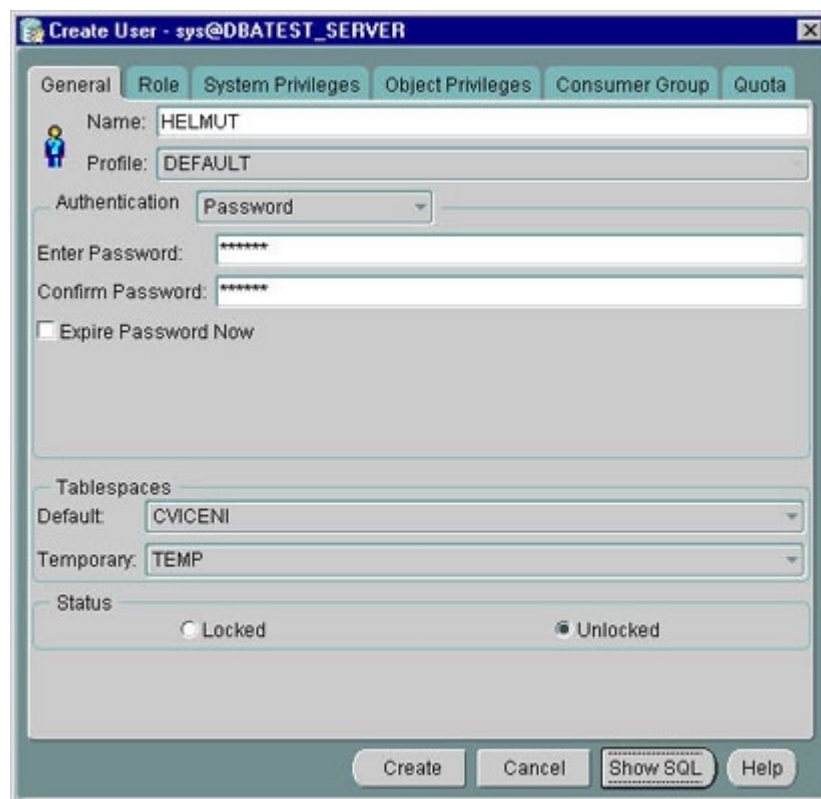
1. Z nabídky objektů vybereme *Tablespace* a pokračujeme ⇒ *Create*
2. Vyplníme jméno – *Name: CVICENI* (to bude zároveň implicitní jméno datového souboru v uvedeném adresáři – *CVICENI.ora*)  
Velikost souboru (*Size*) postačí: 2 MB (v případě naplnění této velikosti se bude datový soubor automaticky zvětšovat)
3. Zvolíme ⇒ *Create* a potvrdíme informační hlášení *OK*

### SQL kód:

```
CREATE TABLESPACE "CVICENI "  
LOGGING DATAFILE 'D:\ORACLE\ORADATA\DBATEST\CVICENI.ora '  
SIZE 2M EXTENT  
MANAGEMENT LOCAL;
```

### 3.3.3 Nový uživatel resp. schéma

Nového uživatele (resp. schéma) vytvoříme s využitím nástroje **Security Manager** – s přihlášením k databázi stále jako uživatel SYS.



Obr. 3.3.3: Vytvoření nového uživatele

1. Z nabídky *Object* ⇒ *Create* vybereme *User* a pokračujeme ⇒ *Create*
2. Vyplníme jméno uživatele *Name*: HELMUT a 2x heslo – *Enter* resp. *Confirm Password* (také HELMUT)
3. V poli *Tablespaces* vybereme jako *Default* v předchozí kapitole vytvořený tabulkový prostor *CVICENI*
4. Na druhé záložce s názvem *Role* přiřadíme uživateli, kromě standardní role *CONNECT*, ještě *DBA* a *RESOURCE* – viz. popis níže. Výběr provedeme přesunem šipkou z horního seznamu do spodní části okna.
5. Zvolíme ⇒ *Create* a potvrdíme informační hlášení *OK*

### SQL kód:

```
CREATE USER "HELMUT"                                //vytvoření uživatele
PROFILE "DEFAULT"
IDENTIFIED BY "helmut"
DEFAULT TABLESPACE "CVICENI" TEMPORARY TABLESPACE "TEMP"
ACCOUNT UNLOCK;

GRANT "CONNECT" TO "HELMUT";                       //přiřazení rolí uživateli
GRANT "DBA" TO "HELMUT";
GRANT "RESOURCE" TO "HELMUT";
```

### Hlavní předinstalované role

<b>CONNECT</b>	Umožňuje připojování k databázi.
<b>RESOURCE</b>	Umožňuje vytváření objektů – tabulky, pohledy atd.
<b>DBA</b>	Umožňuje administrátorské funkce jako např. vytváření uživatelů nebo změna struktury tabulek.

### 3.3.4 Vytvoření tabulek

Tabulky vytvoříme pomocí nástroje **Schema Manager** – nyní už pod novým uživatelským jménem a heslem (*helmut / helmut*). Od této chvíle budou vytvořené objekty zahrnuté (a identifikovány) ve schématu *HELMUT*.

Z menu opět zvolíme *Object* ⇒ *Create* a vybereme *Table*. Na spodní straně okna je možnost zaškrtnout *Use Wizard*, což proces vytvoření tabulky rozdělí do jednotlivých kroků – každý v samostatném okně. Bez použití této volby jsou všechny přístupny v jednom okně s více záložkami. Tento druhý způsob bude nyní popsán.

#### Tabulka STUDENTI

1. Jméno tabulky – *Name*: STUDENTI
2. Vybereme *Schema*: HELMUT
3. Vybereme *Tablespace*: CVICENI
4. V poli *Define Columns* nastavíme sloupce – viz. kapitola 3.1  
Jméno sloupce (*Name*), Datový typ (*Datatype*) + příslušné parametry  
U sloupce ID zaškrtneme v pole *Nulls?* – musí vždy obsahovat hodnotu
5. V záložce *Constraints* nastavíme sloupec ID jako primární klíč (*primary*) – viz. kapitola 1.1
6. Zvolíme *Create* a potvrdíme vytvoření *OK*

## SQL kód

```
CREATE TABLE "HELMUT"."STUDENTI" (           //vytvoření tabulky
"JMENO" VARCHAR2(15),
"PRIJMENI" VARCHAR2(20),
"ID" NUMBER(2) NOT NULL,
"ROCNIK" NUMBER(1),
"KRUH" NUMBER(2),
"EMAIL" VARCHAR2(40),
PRIMARY KEY("ID"))
TABLESPACE "CVICENI" ;
```

## Tabulka TESTY

Podobně jako u předchozí tabulky:

1. Jméno tabulky – *Name*: TESTY
2. Vybereme *Schema*: HELMUT
3. Vybereme *Tablespace*: CVICENI
4. V poli Define Columns nastavíme sloupce – viz. kapitola 3.1  
Jméno sloupce (*Name*), Datový typ (*Datatype*) + příslušné parametry  
U sloupce ID „zakážeme“ hodnoty *Null* a navíc u testů nastavíme implicitní hodnotu (*Default Value*) na „0“ – pro případné pozdější použití<sup>15</sup>.
5. V záložce *Constraints* nastavíme sloupec ID jako cizí klíč (*foreign*) – odkaz na sloupec ID v tabulce STUDENTI.  
*Name*: CIZI (libovolné)  
*Type*: FOREIGN  
*Referenced Schema*: HELMUT  
*Referenced Table*: STUDENTI  
*Table Columns* i *Referenced Columns*: ID
6. Zvolíme *Create* a potvrdíme vytvoření OK

## SQL kód

```
CREATE TABLE "HELMUT"."TESTY" (           //vytvoření tabulky
"ID" NUMBER(2) NOT NULL,
"TEST1" NUMBER(1) DEFAULT 0,
"TEST2" NUMBER(1) DEFAULT 0,
"TEST3" NUMBER(1) DEFAULT 0,
"TEST4" NUMBER(1) DEFAULT 0,
```

---

<sup>15</sup> Hodnota NULL je neznámá hodnota (např. NULL + 5 = NULL). Bez nastavení implicitní hodnoty by se při výpočtech musela hodnota NULL převádět „na nulu“ pomocí speciální funkce NVL – např. NVL(TEST1, 0).

```
CONSTRAINT "CIZI" FOREIGN KEY("ID")
REFERENCES "HELMUT"."STUDENTI"("ID")
TABLESPACE "CVICENI";
```

## 3.2 Vstup dat

Tabulka STUDENTI bude naplněna údaji několika (cca 10) studentů a tabulka TESTY smyšlenými hodnotami výsledků testů.

Nové hodnoty do tabulky je možno zadat buď SQL příkazy (*INSERT*) nebo pomocí utility *Table Data Editor* v rámci *Schema Manageru*.

Údaje budou zadávány bez diakritiky, neboť v tomto případě nebyla k dispozici podpora znaků české národní abecedy.

### Vstup pomocí SQL příkazů

Pro práci s SQL příkazy je možno použít *SQL Plus* (serverová komponenta – viz. kapitola 2.2.1) nebo lépe *SQL\*Plus Worksheet*, který poskytuje pohodlné uživatelské rozhraní (viz. kapitola 2.2.3).

Po spuštění *SQL\*Plus Worksheet* se přihlásíme jménem a heslem (helmut/helmut) + zadáme *Service Name* (dbatest\_server).

Všechny příkazy musí končit středníkem a „spouští“ se klávesou *F5* nebo ikonou *Execute*.

#### Přidání záznamu:

```
INSERT INTO studenti VALUES
('Ales', 'Burghardt', '1', '5', '62', 'burgy@email.cz');
```

Jména jednotlivých sloupců není třeba zadávat, pokud přidáváme celý řádek a uvedené hodnoty jsou v příslušném pořadí.

#### Přidání potvrzeno:

```
1 row created.
```

Takto můžeme přidat i ostatní řádky.

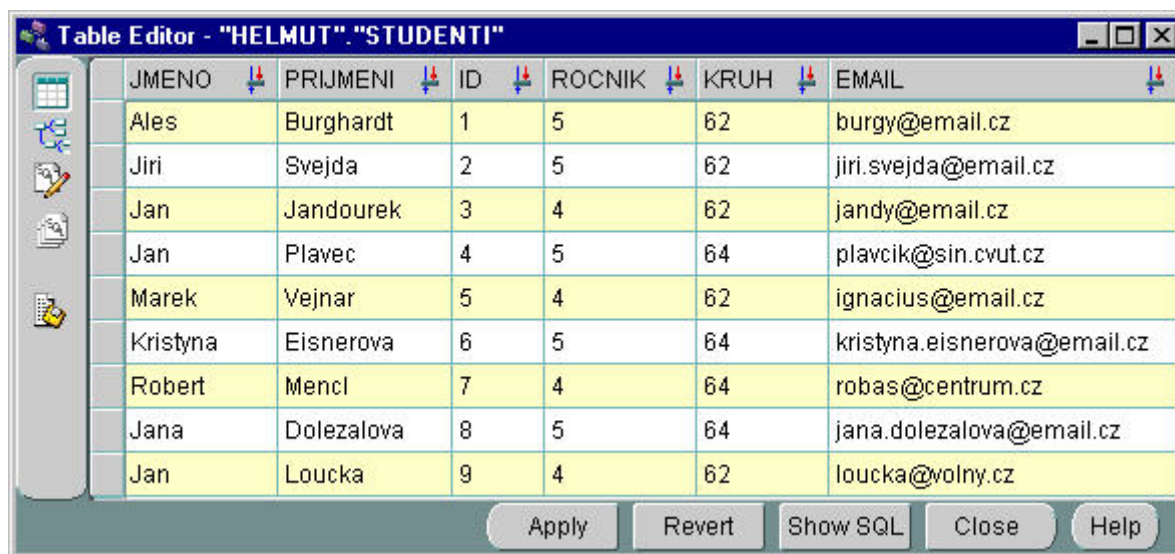
#### Potvrzení změn v tabulce:

Všechny prováděné změny (dohromady jedna *transakce*) se ukládají do odkládacího prostoru (*Rollback Segment*). Aby se všechny změny uložily do databáze, musí se potvrdit příkazem *COMMIT*. Stornování změn se provede příkazem *ROLLBACK*.

```
Commit complete.
```

## Table Data Editor

*Table Data Editor* je nástroj v rámci *Schema Manageru*, který umožňuje v grafické podobě editovat (přidávat, mazat, měnit) a třídít data v tabulce. Zároveň obsahuje i *Graphical Select Mode*, který dovoluje tabulku dotazovat (příkaz SELECT) v grafické i klasické podobě. Další funkcí tohoto nástroje je *Save List*, která umožňuje tabulku, nebo zvolené sloupce, uložit do souboru jako tabulku HTML<sup>16</sup>, Text nebo hodnoty oddělené čárkou (pro tabulkové procesory jako MS Excel).



JMENO	PRIJMENI	ID	ROCNIK	KRUH	EMAIL
Ales	Burghardt	1	5	62	burgy@email.cz
Jiri	Svejda	2	5	62	jiri.svejda@email.cz
Jan	Jandourek	3	4	62	jandy@email.cz
Jan	Plavec	4	5	64	plavcik@sin.cvut.cz
Marek	Vejnar	5	4	62	ignacius@email.cz
Kristyna	Eisnerova	6	5	64	kristyna.eisnerova@email.cz
Robert	Mencl	7	4	64	robass@centrum.cz
Jana	Dolezalova	8	5	64	jana.dolezalova@email.cz
Jan	Loucka	9	4	62	loucka@volny.cz

Obr. 3.2: Table Data Editor

Přihlásíme se do *DBA Studia* – viz. kapitola 3.3.1 a při zaškrtnuté volbě v menu *View* ⇒ *By Schema* nalistujeme ve stromové struktuře požadovanou tabulku – *Schema* ⇒ *HELMUT* ⇒ *Tables* ⇒ *STUDENTI*. Na této tabulce klikneme pravým tlačítkem myši a zvolíme *Table Data Editor*.

Nyní lze hodnoty zadávat přímo do polí v podobě listu tabulkového procesoru a po provedení potvrdit transakci ⇒ *Apply*.

Stejným způsobem naplníme daty i tabulku *TESTY*.

<sup>16</sup> HTML (HyperText Markup Language) je jazyk používaný pro vytvoření webových stránek.



## 4. Možnosti prezentace a správy dat na Internetu

### 4.1 Úvod

Jak již bylo řečeno, databázové technologie jsou široce rozšířeny a jejich vzestup stále pokračuje. S tím vyvstává požadavek na to, aby data byla zpřístupněna v co nejširším měřítku. Prostředkem, který toto spojení zajišťuje je „všudypřítomný“ Internet, resp. World Wide Web - WWW. Zpočátku se jednalo jen o stránky s neměnným obsahem, ale dnes se odhaduje, že přibližně polovina stránek je generována dynamicky [14] – tj. obsah většiny z nich se vytváří na základě informací poskytovaných nějakým databázovým systémem. Uživatel tak může komunikovat s databázovou aplikací jen prostřednictvím webového prohlížeče.

Oracle8i standardně obsahuje komponentu *Web Publishing Assistant* s jehož pomocí je možno vytvářet „pseudodynamické“ webové stránky na základě SQL dotazů. Dalším nástrojem, už samostatně dodávaným, je *Oracle WebDB*, což je kompletní vývojové prostředí nabízející řešení, jak vytvářet dynamické webové stránky založené na datech z databáze. Tyto produkty a zejména WebDB budou popsány v následujících kapitolách.

### 4.2 Web Publishing Assistant

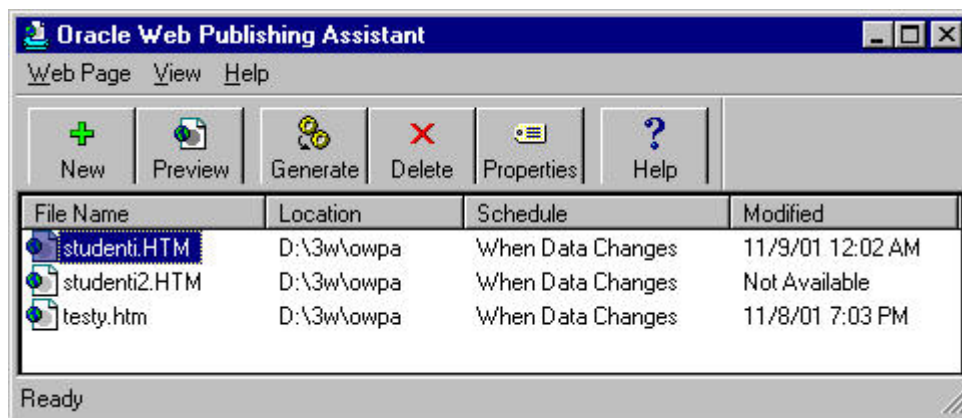
#### 4.2.1 Princip

Tento asistent umožňuje vytvářet sestavy ve formátu HTML na základě databázových tabulek nebo konkrétních SQL dotazů. Takto vygenerované webové stránky je možno v pravidelných intervalech obnovovat a udržovat je tak v aktuálním stavu. Možnosti obnovování budou popsány níže.

Výsledky můžeme formátovat s využitím přednastavených šablon (*templates.HTX*) (více viz. kapitola 4.2.3).

Práce s ním je velmi jednoduchá a v několika minutách se dají požadovaná data připravit ke zveřejnění na Internetu. To vše bez nutnosti programovat v Javě nebo HTML. Navíc možnosti nastavení obnovování jsou velmi široké – stránky se mohou obnovovat i po několika minutách a tím se již v principu blíží dynamickým.

Aby obnovování stránek probíhalo, musí být aktivní služba (pod Windows NT) *OracleWebAssistant* – viz. *Start* ⇒ *Settings* ⇒ *Control Panel* ⇒ *Services*.



Obr. 4.2.1: Oracle Web Publishing Assistant

## 4.2.2 Vytvoření webové stránky

Proces vytvoření nové webové stránky je rozdělen do čtyř základních kroků – každý v samostatném okně. Spustí se tlačítkem *New* v nástrojové liště nebo *Create Web Page* z uvítací stránky:

### 1. Přihlášení k databázi (*Login*)

Zadává se uživatelské jméno / heslo (*User Name / Password*) a jméno databáze (resp. služby – viz. kapitola 3.3.1)

### 2. Výběr dat pro zveřejnění (*Query*)

První možností výběru je grafická identifikace tabulky nebo jednotlivých sloupců ve stromové struktuře objektů v databázi.

Druhý způsob je přímé zadání dotazu v SQL (popřípadě jeho import ze souboru) – výsledek tohoto dotazu bude použit pro webovou stránku.

### 3. Nastavení obnovování (*Scheduling Options*)

Můžeme vybírat z následujících možností:

***Immediately*** – vytvoří webovou stránku okamžitě bez možnosti budoucího obnovení

***Once*** – nastavení roku, měsíce, dne a času v příštích 31 dnech, kdy se stránka (jednou) vytvoří

***On Day(s) of the Week*** – dovoluje nastavit čas a den (nebo dny) v týdnu, kdy se stránka bude vytvářet (např. Pondělí, Středa, Pátek v 1:00 hod.)

***On Date(s) of the Month*** – pro vytvoření stránky lze nastavit čas a libovolný počet dnů v měsíci (např. 1., 15. a 30. den v 0:00 hod.)

***At Regular Intervals*** – dovoluje nastavit interval obnovování. Stránka se vytvoří okamžitě a od té chvíle v určený interval. Ten musí být v intervalu

jedné minuty až 31 dní.

**When Data Changes** – stránka se vytvoří a dále obnoví vždy při změně dat v databázi. Tato volba je možná jen, když při výběru dat byl použit první způsob (viz. bod 2).

**Manually** – při této volbě bude obnovování spouštěno „ručně“. Spustí se příkazem *Generate* v nástrojové liště Web Assistent – viz. obrázek 4.2.1.

#### 4. Jméno souboru a formátování (*File Options*)

Nastavuje se jméno resp. umístění souboru ve formátu HTML a dále formátování vystupujících dat prostřednictvím šablony (*Template.htm*). K dispozici je standardní šablona (*Generate Default Template*) nebo můžeme použít vlastní.

Stránka se definitivně vytvoří zvolením *Finish*.

### 4.2.3 Šablony

Šablony umožňují formátovat výstupy dat do webové stránky. Jedná se o soubory s příponou HTX. Obsahují HTML kód doplněný o *klíčová slova*, která jsou podobná standardním HTML tagům<sup>17</sup> ve tvaru `<%KEYWORD%>`. Web Publishing Assistant při použití šablony (viz. bod 4 předchozí kapitoly) vygeneruje běžný HTML soubor (standardně *web.htm*).

Implicitní (*default*) šablona využívá k formátování dat Javovský applet<sup>18</sup>. Jeho výhodou je hlavně nezávislost na platformě (operačním systému i hardwaru) a webovém browseru (různé browsery obsahují odlišnosti v interpretaci HTML kódu). Šablony je možno upravovat (měnit barvu pozadí, textu, hlavičky atd.) přímo ve zdrojovém kódu nebo v libovolném HTML editoru – to vše i je-li už stránka vytvořena (v závislosti na nastavení obnovování se příště již vygeneruje podle změněné šablony v nové podobě). Podrobnější informace viz. [18].

#### **Klíčová slova:**

**`<%begindetail%>`, `<%enddetail%>`**

Tato klíčová slova ohraničují oblast, která se generuje cyklicky pro každý datový záznam. Jednotlivé hodnoty každého záznamu se umísťují prostřednictvím klíčových slov ve tvaru `<%jméno_sloupce%>`.

---

<sup>17</sup> Tag je značka HTML ohraničená ostrými závorkami `<začátek_tagu>` `</konec_tagu>`.

<sup>18</sup> Applet je malý program nebo aplikace napsaná v Javě.

**<%if podmínka%>, <%else%>, <%endif%>**

Tímto způsobem můžeme do šablony vložit podmínku. Používá se např. pro případ vypsaní hlášení pokud zadanému dotazu nevyhoví ani jeden záznam. V podmínkách se používají operátory **EQ** (=), **LT** (<), **GT** (>), **CONTAINS** (obsahuje).

**Proměnné:**

**<%CurrentRecord%>** - obsahuje číslo, které udává kolikátý záznam se právě vypisuje. To je kolikrát proběhl „cyklus *begindetail – enddetail*“. Při prvním průběhu má hodnotu nula (0).

**<%CurrentDate%>** - obsahuje datum a čas, kdy byla stránka vygenerována. Formát lze měnit množstvím parametrů – viz. [18].

**<%MaxColumn%>** - udává počet sloupců ve výsledku daného dotazu.

**<%QueryId%>** - obsahuje unikátní číslo vygenerované stránky (dotazu).

#### 4.2.4 Konkrétní příklad

Prakticky bude funkce Web Publishing Asistenta ukázána na tabulkách STUDENTI a TESTY. Bude vytvořena webová stránka s číslem, jménem, příjmením studenta a výsledky všech testů – viz. bod 2. Postup odpovídá bodům z kapitoly 4.2.2 .

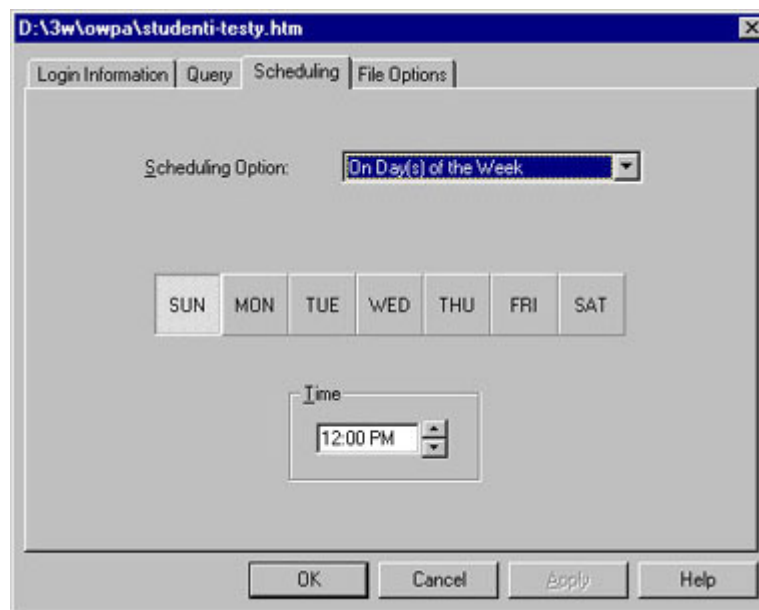
1. Přihlášení: jméno / heslo (*helmut / helmut*) a jméno databáze (*dbatest\_server*)
2. Výběr dat provedeme pomocí SQL dotazu (volba *Enter a query as a SQL statement*). Bude vybráno celkem 7 sloupců, přičemž sloupce Jméno a Příjmení budou ve výstupu spojeny<sup>19</sup> v jeden.

```
SELECT studenti.id, prijmeni || ' ' || jmeno AS "Jméno",
test1, test2, test3, test4
FROM studenti, testy
WHERE studenti.id = testy.id
ORDER BY prijmeni;
```

3. Obnovování nastavíme na každou neděli v půlnoci – na kartě *On Day(s) of the Week*, zvolíme neděli (SUN) a čas 12:00 PM – viz. obrázek 4.2.4 – 1.
4. Zadáme jméno a plnou cestu k souboru. Dále vybereme šablonu, podle které se bude webová stránka formátovat. Tato šablona již byla vytvořena předem – viz. příloha 9.1 a kapitola 4.2.3. Nakonec zvolíme *Finish* a potvrdíme *OK*.

---

<sup>19</sup> Ke spojování řetězců v SQL používá systém Oracle dvojnák „||“ (CONCATENATE).



Obr. 4.2.4 – 1: Nastavení obnovování webové stránky  
(On Day(s) of the Week – Neděle, 0:00 hod.)

ID	Jméno	Test 1	Test 2	Test 3	Test 4
1	Burghardt Ales	2	1	2	1
8	Dolezalova Jana	2	2	3	4
6	Eisnerová Kristyna	3	2	2	2
3	Jandourek Jan	3	3	2	4
9	Loucka Jan	3	4	2	3
7	Mencl Robert	3	4	2	1
4	Plavec Jan	4	4	3	4
2	Svejda Jiri	2	2	3	2
5	Vejnár Marek	3	4	2	4

Obr. 4.2.4 – 2: Webová stránka vytvořená pomocí Oracle Web Publishing Assistant

## 4.3 Oracle WebDB

### 4.3.1 Úvodní popis

Oracle WebDB je kompletní řešení pro vytváření, rozvíjení a monitorování webových databázových aplikací a webů obsahově závislých na databázi [19]. Dále umožňuje spravovat databázové objekty (tabulky, pohledy, funkce, uživatele, role, práva atd.), vytvářet prvky HTML rozhraní (formuláře, grafy, menu a reporty<sup>20</sup>) nebo nastavovat web server. To vše přes jednoduché rozhraní založené na HTML s použitím standardního webového prohlížeče (bez znalosti HTML a Java).

Výsledná aplikace tedy spojuje výhody Internetu (snadný přístup k informacím) a databáze<sup>21</sup> Oracle (vysoká úroveň zabezpečení proti neautorizovanému přístupu).

Produkt Oracle WebDB byl v dalších verzích přejmenován na *Oracle Portal*<sup>22</sup>.

WebDB je založena na třívrstevném modelu [14]. Aplikaci není nutné instalovat a spravovat v mnoha kopiích na každém počítači, ale stačí pouze jednou na aplikačním severu. Díky tomu může být klientem libovolné zařízení vybavené pouze webovým prohlížečem (první vrstva). Druhou vrstvou představuje web server. Tím může být standardně dodávaný *Oracle WebDB Listener* (pod Windows NT musí „běžet“ stejnojmenná služba – viz. Dodatek B) nebo libovolný *HTTP*<sup>23</sup> server podporující rozhraní *CGI*<sup>24</sup> (např. *Microsoft IIS* nebo *Apache*). Třetí vrstvou je databázový server Oracle 8i, 8.0.5 nebo 7.3.4 (s některými omezeními).

Uživatelské rozhraní WebDB je představováno sadou dynamicky generovaných HTML stránek s Java skriptem – obsahuje jen čistý HTML kód<sup>25</sup> bez přítomnosti Javy. Proto je možné použít na straně klienta i méně výkonný počítač – veškerá řídicí a kontrolní činnost je přesunuta na aplikační server, kde je nainstalována WebDB.

---

<sup>20</sup> Reporty poskytují výsledky SQL dotazů ve formě tabulky.

<sup>21</sup> Vlastní data mohou být uložena v libovolné relační databázi [29].

<sup>22</sup> Všechny změny ve jménech produktů Oracle Corporation jsou uvedeny např. na <http://programovanie.pc.sk/databazy/clanok.ltc?ID=151>

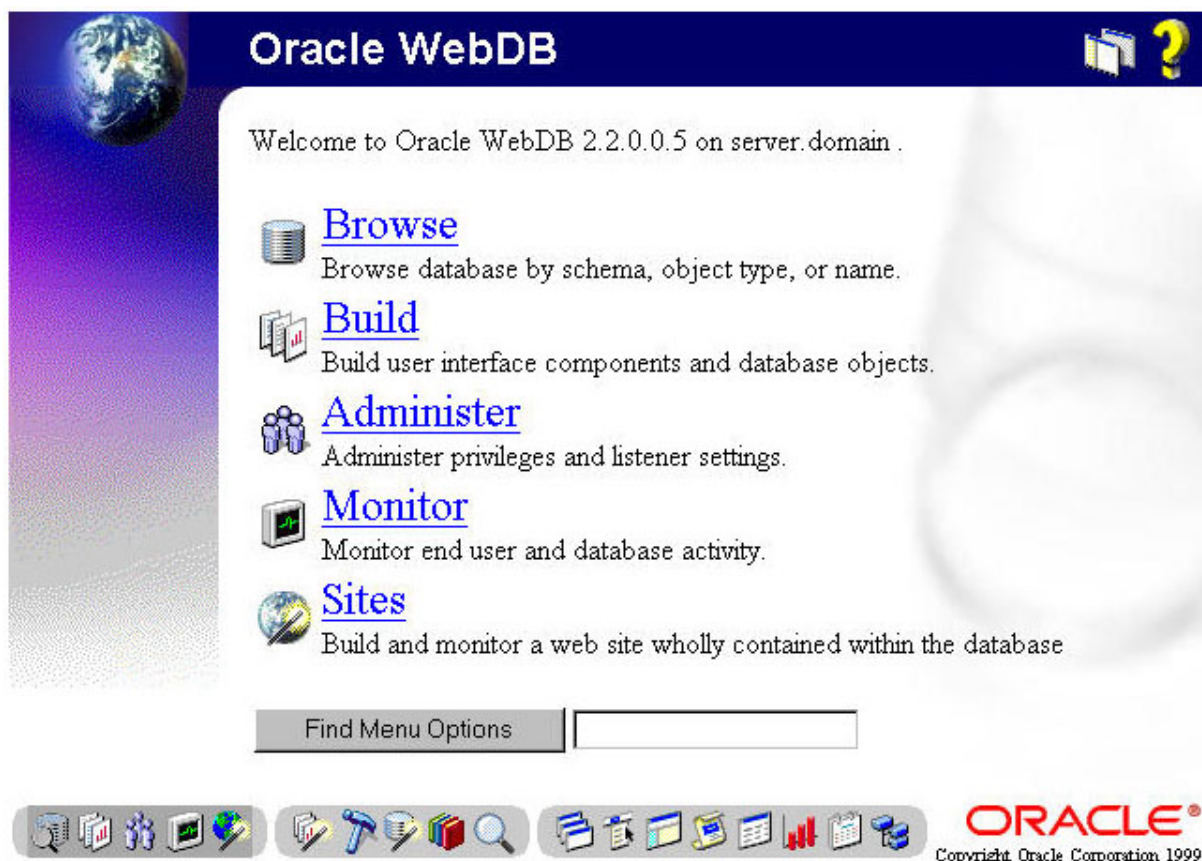
<sup>23</sup> HTTP (HyperText Transfer Protocol) je [2] protokol zajišťující přenos HTML stránek z WWW serveru do prohlížeče.

<sup>24</sup> CGI (Common Gateway Interface) je [2] rozhraní definující způsob spouštění programů a předávání dat mezi WWW-serverem a programem.

<sup>25</sup> Další vývoj směřuje ke generování výstupu v XML, což umožní provoz aplikací i na speciálních zařízeních jako jsou například mobilní telefony [11].

### 4.3.2 Struktura a funkce WebDB

Po úspěšném vstupu do WebDB (viz. dále kapitola 4.3.3) se objeví úvodní obrazovka s hlavním menu, které obsahuje pět položek pokrývajících všechny základní funkce celé aplikace. Dále sadu ikon pro rychlejší přístup k některým funkcím a pole pro vyhledávání v položkách menu.



Obr. 4.3.2: Úvodní strana Oracle WebDB

**Stručný přehled funkcí v jednotlivých položkách (viz. Obr. 4.3.2):**

#### **Browse**

Tato volba poskytuje grafický přehled objektů v databázi (tabulky, pohledy, synonyma, funkce atd.). Lze je třídit podle jména, typu objektu nebo schémat (vlastnictví). Jakmile najdeme konkrétní objekt, můžeme s ním (podle jeho typu) provádět např.:

- dotazovat, prohlížet nebo aktualizovat obsah tabulek
- spouštět procedury a funkce
- zjišťovat informace o synonymech nebo indexech

Uživatel je oprávněn prohlížet komponenty jen ve schématu, ve kterém má právo prohlížení (*Browse In* privileges).

## Build

Zde jsou soustředěny nástroje pro vytváření a editaci jednotlivých komponent<sup>26</sup> a objektů. K dispozici jsou průvodci, kteří krok po kroku vedou uživatele procesem vytvoření jednotlivých komponent. Uživatel může vytvářet komponenty jen ve schématu, ve kterém má oprávnění (*Build In* privileges).

Všechny funkce jsou rozděleny do následujících podskupin:

- **User Interface Components** – nástroje pro vytváření prvků uživatelského (webového) rozhraní:
  - *Forms* – formuláře poskytují rozhraní pro úpravu dat v tabulce
  - *Menus* – menu sestavené z hypertextových odkazů (viz. dále)
  - *Frame Drivers* – rozdělení stránky na více částí pomocí rámců (frames)
  - *Dynamic Pages* – vytváření dynamických stránek pomocí speciálního tagu <ORACLE>, do kterého je možno přímo vkládat PL/SQL kód
  - *Reports* – výstupy hodnot databázových tabulek a pohledů
  - *Charts* – vytváření grafů založených na datech databázové tabulky
  - *Calendars* – zobrazení výsledku SQL dotazu ve formě kalendáře
  - *Hierarchies* – vytváření hierarchických modelů
- **Shared Components** – vytváření sdílených komponent dostupných i ostatním uživatelům k opakovanému použití. Jsou to seznamy barev (*Colors*), obrázky (*Images*), hypertextové odkazy (*Links*), uživatelské šablony (*User Interface Templates*), seznamy písem (*Fonts*), skripty v jazyce *JavaScript* a seznamy hodnot pro výběr z omezené množiny (*Lists of Values – LOV*).
- **Utilities** – obsahuje nástroj pro export sdílených komponent (*Export Shared Components*). Používá se pro přenos informací z tabulky sdílených komponent např. na jinou instalaci WebDB ve formě vygenerovaného SQL kódu (INSERT). Druhou funkcí je *Manage Locks on Components*, která umožňuje monitorovat editování jednotlivých komponent (edituje-li někdo komponentu, pro ostatní uživatele je editace zamezena).
- **Database Objects** – sestává z nástrojů pro tvorbu databázových objektů jako funkce, procedury, tabulky, synonyma, pohledy apod.

---

<sup>26</sup> WebDB odlišuje termín komponenty (components) a objekty (objects). Komponenty jsou dynamicky generované webové stránky a obsah, který vytvoří WebDB. Objekty se rozumí objekty databáze Oracle, na kterých jsou komponenty založeny.



- **Find Components** – umožňuje vyhledávat a třídit existující komponenty podle jména, schématu nebo typu.

## Administer

V této sekci najdeme nástroje pro správu uživatelů, rolí a práv – včetně specifických oprávnění *Browse In* a *Build In* (viz. výše).

- **User Manager** – pomocí tohoto nástroje lze vytvářet nové uživatele nebo vyhledávat a měnit parametry stávajících. Měnit lze všechny standardní vlastnosti jako heslo nebo role a navíc oprávnění *Browse In* a *Build In*.
- **Grant Manager** – umožňuje přiřazovat či odebírat oprávnění uživatelům resp. rolím na databázové objekty. Klasickým způsobem (viz. výše) můžeme vyhledat objekt a podle jeho typu mu přiřadit nebo odebrat oprávnění na dotazování (SELECT), vkládání (INSERT), změnu (UPDATE), mazání (DELETE) nebo spouštění (EXECUTE).
- **Role Manager** – slouží k vytváření nových rolí nebo k vyhledávání a správě stávajících (přiřazování či odebírání uživatelů).
- **Change Your Password** – umožňuje změnit heslo aktuálního uživatele („vlastní heslo“).
- **Report WebDB Privileges** – poskytuje přehled uživatelů, kteří mají přiřazenu roli WEBDB\_DEVELOPER (viz. dále) a oprávnění *Browse In* a *Build In*.
- **Configure WebDB Activity Log** – umožňuje nastavit dobu platnosti a prohlížet tabulku záznamů o „žádostech“ uživatele na objekty (obsahuje přehled objektů a dobu jejich první a poslední aktivity atd. viz. *Monitor*).
- **Listener Settings** – dovoluje spravovat vstupní parametry (viz. více v kapitole 4.3.3) a webový server (*Oracle WebDB Listener*).

## Monitor

Obsahuje nástroje pro monitorování aktivit uživatelů a samotné databáze. Tak lze vyhodnocovat zatížení systému a výkon jednotlivých komponent.

- **User Interface Components** – ve formě grafů poskytuje informace o využití komponent jako je doba odezvy a počet záznamů ve výsledku, jmenný seznam komponent, denní a hodinové využití, seznam koncových uživatelů včetně jejich IP adresy a typu browseru.
- **Browse Activity Log** – umožňuje procházet (hledat, třídit apod.) tabulku se záznamy o využití komponent – viz. předchozí bod.

- **Database Objects** – obsahuje nástroje pro sledování parametrů a aktivit samotné Oracle databáze jako např. informace o databázi a objektech v ní, spotřebě paměti, statistiky využití prostoru datovými soubory atd.

## Sites

Tato položka, prostřednictvím nástroje *Site Builder*, slouží k logickému uspořádání informací do ucelených webových sídel (*Websites*). Součástí této struktury mohou být existující komponenty (reporty, formuláře apod.), ale i libovolné dokumenty uložené přímo v databázi. Po vložení nových informací či celého dokumentu je tento automaticky indexován pro vyhledávací mechanismus. Před publikováním lze požadovat schválení nebo nastavit dobu, po kterou bude dokument přístupný. Při změně obsahu je možno uchovávat i jednotlivé verze.

Při tvorbě designu website lze využívat všechny standardní prvky jako obrázky, menu z hypertextových odkazů či rámce (frames). Správa lze provádět z prostředí website samotné.

Obsahově je website rozdělena do tzv. *folderů* (obvykle položky hlavního menu), přičemž pro každý je možno nastavit individuální přístupová práva. Některé informace tak mohou být veřejně přístupné (*Public*) a jiné vyžadovat přihlášení uživatelským jménem a heslem.

### 4.3.3 Práce s WebDB

#### Přihlášení do systému

Ve webovém browseru zadáme adresu<sup>27</sup> pro přístup na úvodní stranu WebDB (resp. WebSite), která se skládá ze tří částí:

`jméno_serveru.doména:port/Database_Access_Descriptor`

<b>Část adresy</b>	<b>Popis</b>	<b>„Naše“ hodnota</b>
Jméno serveru	Jméno serveru, na kterém je nainstalována WebDB	SERVER
Doména	Síťová doména serveru	DOMAIN
Port	Číslo portu pro přístup k webserveru (je-li port 80, může být vynechán).	(80)
Database Access Descriptor (DAD)	Hodnota sloužící jako vstupní brána do databáze. Je různá pro každou instanci <sup>28</sup> WebDB nebo WebSite v databázi.	WebDB

<sup>27</sup> Máme-li všechny 3 vrstvy systému zprovozněny na jednom počítači (jako v tomto případě), stačí zadat doménové jméno „localhost“ nebo IP adresu 127.0.0.1

<sup>28</sup> Instancí rozumíme jednu konkrétní instalaci WebDB pod určitým uživatelským jménem.

Poté se objeví okno pro zadání uživatelského jména a hesla – po úspěšném přihlášení se objeví úvodní obrazovka WebDB (viz. obr. 4.3.2). Uživatelský účet je týž jako v databázi Oracle, na které je WebDB nainstalována. Pokud *DAD* ukazuje na veřejně přístupnou část WebSite, přihlášení je vynecháno.

### Pohyb uvnitř WebDB

Pro pohyb ve struktuře WebDB a dosažení jednotlivých funkcí jsou k dispozici následujícími způsoby:

- Listování v hlavních kategoriích – viz. kapitola 4.3.2
- Grafická nástrojová lišta pro hlavní funkce, která je na spodní straně každé stránky WebDB







- Historie odkazů – při procházení jednotlivých stránek se ve spodní části okna tvoří postupná historie odkazů, kterými jsme do té doby prošli

*Back:* [User Interface Components](#), [Build](#), [Oracle WebDB](#)

- Formulářové pole *Find Menu Options* na hlavní straně WebDB umožňuje vyhledávat text v položkách menu a jeho popiscích




- Pohled s rámci – kliknutím na ikonu  v pravém horním rohu okna se obrazovka rozdělí na dvě části, přičemž v levé je stále k dispozici hlavní struktura WebDB a vyhledávací pole – v pravé části se zobrazuje obsah

Z jakéhokoli místa se lze vrátit na úvodní stranu ikonou  v pravém horním rohu okna. Mezi jednotlivými stránkami lze přecházet i pomocí tlačítek *Vpřed* (*Forward*), *Vzad* (*Back*) webového prohlížeče, avšak ne v průběhu vytváření objektů pomocí průvodců (*Wizards*). V takovém případě je nutno používat šipky  a .

### Role WEBDB\_DEVELOPER

Aby uživatel WebDB mohl vytvářet komponenty a databázové objekty, musí mít přiřazenu roli WEBDB\_DEVELOPER. Proto ji nyní přiřadíme uživateli HELMUT.

1. Na hlavní stránce WebDB zvolíme *Administer* (popřípadě  v nástrojové liště) a dále *User Manager*.
2. V poli *Select Recently Created Users* vybereme uživatele HELMUT

3. Na kartě *Roles* v horní části nalistujeme roli `WEBDB_DEVELOPER` a přidáme ji k ostatním tlačítkem *Add*. Změnu uložíme tlačítkem *Apply*.

Obdobným způsobem můžeme na kartě *Browse Privileges* resp. *Build Privileges* nastavit oprávnění, ve kterých schématech může uživatel prohlížet resp. vytvářet WebDB komponenty a objekty.

Roli lze rovněž uživateli přiřadit standardními administrátorskými nástroji (nebo SQL příkazem) v prostředí Oracle8i (viz. kapitola 3).

### Správa komponent

Všechny vytvořené komponenty lze i po vytvoření kdykoli spravovat.



Obr. 4.3.3: Nástroje pro správu jednotlivých komponent WebDB

- **Edit** – změny všech parametrů zvolených při vytvoření
- **Run** – spuštění (zobrazení)
- **Parameters** – volba vstupních parametrů
- **Privileges** – nastavení uživatelů (nebo rolí), kteří mají právo komponentu spouštět (*Execute*)
- **Monitor** – statistika využívání komponenty
- **Manage** – obsahuje nástroje pro mazání, export, přejmenování nebo kopírování komponent

K zobrazení komponenty ve webovém prohlížeči slouží adresa ve tvaru:

`http://jméno_serveru.doména/DAD/SCHÉMA.KOMPONENTA.show`

### Odhlášení a informace o uživateli

Kliknutím na symbol zeměkoule v levém horním rohu okna se zobrazí stránka s informacemi o aktuálním uživateli. Jsou to mimo jiné identifikace uživatele, jeho IP adresa, typ prohlížeče, přidělené role apod.



Dále je zde odkaz na funkci změna hesla (*Change Password*) a odhlášení ze systému (**Log off**).

#### 4.3.4 Vytvoření ukázkové aplikace


Tvorba ucelené aplikace pomocí WebDB spočívá ve vytvoření jednotlivých komponent na základě dat v databázi a jejich vzájemném propojení. Pro získání dat z databázových tabulek můžeme použít reporty (*Reports*), popřípadě dynamické stránky (*Dynamic Pages*). Editaci, vkládání nebo mazání dat zajistíme prostřednictvím formulářových polí (*Forms*). Pro vzájemné propojení komponent slouží hypertextové odkazy (*Links*). Menu (*Menus*) poslouží k jejich logickému uspořádání a šablony (*User Interface Templates*) k úpravě vzhledu jednotlivých stránek.

Nyní zde budou stručně uvedeny postupy vytvoření základních komponent s běžným nastavením, které jsou použity v ukázkové aplikaci. Seznam a popis jednotlivých komponent je uveden v kapitole 4.3.2. Vzhledem k celkové šíři možností WebDB odkazují například na [19, 20].

##### Report

1. Zvolíme *Build* ⇒ *User Interface Components* ⇒ *Reports* (nebo ikonu  nástrojové lišty).
2. V okně *Create a New Report* zvolíme *Report from Query Wizard* a pokračujeme ⇒ *Create*
3. Vybereme schéma, do kterého bude report patřit a jeho jméno.
4. Vybereme jednu nebo více tabulek se zdrojovými daty pomocí  a *Add*.
5. Zvolíme způsob spojení tabulek (prostřednictvím jakých sloupců).
6. Určíme sloupce, které budou ve výstupu.
7. Podle potřeby zavedeme podmínky pro filtrování záznamů ve výstupu.
8. Nastavíme způsob zobrazení jednotlivých sloupců. V našem případě u prvního sloupce nastavíme navíc předem vytvořený odkaz (*Link*) na formulář pro editaci dat dané tabulky.
9. Nastavíme parametry zobrazení jako barvy, typ písma, maximální počet řádků na výstupu, třídění (*Order by*), typ výstupu (HTML, ASCII, Excel) apod.
10. Nastavení případných vstupních parametrů
11. Volba šablony zobrazení a textu záhlaví, zápatí a titulku
12. Případný PL/SQL kód, který se má vykonat v jednotlivých fázích zobrazování stránky s výsledkem.
13. Potvrdíme vytvoření ⇒ *OK*

## Formulář (Form)

1. Zvolíme *Build* ⇒ *User Interface Components* ⇒ *Forms* (nebo ikonu  nástrojové lišty).
2. V okně *Create a New Form* zvolíme *Forms on Tables/Views* ⇒ *Create*.
3. Vybereme schéma, do kterého bude formulář patřit a jeho jméno.
4. Zvolíme tabulku (nebo pohled), na které bude formulář založen.
5. Vybereme rozvržení *Structured* (druhá volba *Unstructured* dovoluje navrhovat vzhled formuláře přímo pomocí HTML)
6. Nastavíme vzhled celého formuláře a zároveň i jednotlivých jeho polí (pojmenování, výška, šířka, typ písma, barva, formát vkládané hodnoty apod.)
7. Výběr a umístění tlačítek *Insert*, *Update*, *Delete*, *Query* nebo *Reset*
8. Volba šablony zobrazení a textu záhlaví, zápatí a titulku
9. Případný PL/SQL kód, který se má vykonat v jednotlivých fázích zobrazování stránky s výsledkem
10. Potvrdíme vytvoření ⇒ *OK*

## Hypertextový odkaz (Link)

1. Zvolíme *Build* ⇒ *Shared Components* (nebo ikonu  nástrojové lišty) ⇒ *Links*
2. Pokračujeme *Create Link*
3. Vybereme schéma, do kterého bude odkaz patřit a jeho jméno.
4. Z nabídky vybereme *WebDB Component*, na který bude link ukazovat a poté jeho jméno (nalistovat pomocí )
5. Nastavení parametrů zobrazení cíle hypertextového odkazu. Například u formuláře je možno nastavit, které hodnoty v něm budou implicitně uvedeny.
6. Potvrdíme vytvoření ⇒ *OK*

Id	Příjmení	Test 1	Test 2	Test 3	Test 4
<a href="#">1</a>	<a href="#">Burghardt</a>	2	1	2	1
<a href="#">6</a>	<a href="#">Peterka</a>	3	2	2	2
<a href="#">3</a>	<a href="#">Jandourek</a>	3	3	2	4
<a href="#">9</a>	<a href="#">Lacina</a>	3	4	2	3
<a href="#">4</a>	<a href="#">Plavec</a>	4	4	3	4
<a href="#">2</a>	<a href="#">Vachout</a>	2	2	3	2
<a href="#">5</a>	<a href="#">Vejnár</a>	3	4	2	4
<a href="#">8</a>	<a href="#">Rachota</a>	2	2	3	4
<a href="#">7</a>	<a href="#">Mencl</a>	3	4	2	1

Row(s) 1 - 9

Odkazy ve sloupci **ID** - editace znamek

Odkazy ve sloupci **Příjmení** - editace udaju o studentech

Obr. 4.3.4 – 1: Report s výsledky testů

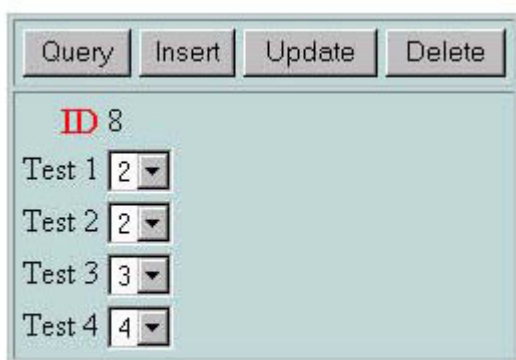
## Výsledná podoba a funkce aplikace

Výchozí stránka prostřednictvím reportu ukazuje výsledky testů. Report je založený na tabulkách STUDENTI, TESTY a zobrazuje pouze sloupce ID, Prijmeni, Test1 – Test4 (obr. 4.3.4 – 1).

Hodnoty ve sloupci *ID* tvoří zároveň odkaz na formulář pro úpravu výsledků testů (obr. 4.3.4 – 2) a ve sloupci *Prijmeni* odkaz na formulář pro úpravu údajů o studentech (obr. 4.3.4 – 3).

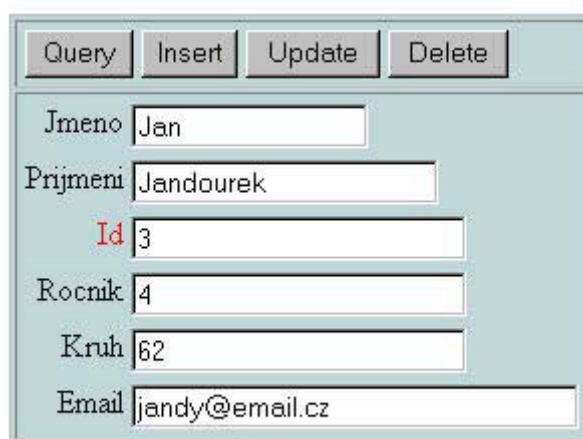
Po kliknutí na odkaz se ve formulářových polích díky parametrům odkazu (*Link*) rovnou zobrazí údaje z příslušného záznamu. Tlačítka umožňují záznamy prohlížet (*Query*), aktualizovat (*Update*), mazat (*Delete*) popřípadě přidávat (*Insert*).

Ve formuláři pro úpravu výsledků testů jsou pole vytvořena ve formě výběru z hodnot 1 – 4 (*List of Values*) a položka *ID* nelze měnit (pouze ve formuláři pro údaje studentů).



[Zpet na vypis dat](#)

Obr. 4.3.4 – 2: Formulář pro správu tabulky Testy




[Zpet na vypis dat](#)

Obr. 4.3.4 – 3: Formulář pro správu tabulky Studenti


U každé komponenty lze individuálně nastavit přístupová práva (viz. správa komponent – *Privileges* v předchozí kapitole). Tímto způsobem lze nastavit report zobrazující výsledky jako veřejně přístupný (*Public*) a formuláře pro úpravu dat ponechat přístupný jen pro konkrétního uživatele (správce).

Úspěšná změna (vlození nebo mazání) dat se projeví příslušným hlášením.


 4 Column(s) Updated.

Column	Old Value	New Value
TEST1	3	2
TEST2	2	3
TEST3	3	2
TEST4	3	2

[Back to form](#)

 1 Row(s) Deleted.

[Back to form](#)

 1 Row Inserted.

Column	Value(s)
JMENO	Rudolf
PRJMENI	Kalina
ID	11
ROCNIK	5
KRUH	64
EMAIL	kaloun@volny.cz

[Back to form](#)

Obr. 4.3.4 – 4, 5, 6: Příklady hlášení po úspěšné manipulaci s daty pomocí formulářů (4 - změna, 5 - mazání, 6 - vložení).

## 4.4 ODBC / JDBC

Pro úplnost je nutno uvést i nejobecnější způsob přístupu k databázi. V roli klientské aplikace může stát libovolný program, který s databází komunikuje prostřednictvím příslušného (nativního) aplikačního rozhraní (*API – Application Programming Interface*) [31]. Každá databáze má svůj vlastní API. Aby byl program schopen komunikovat s více databázemi, musí podporovat více aplikačních rozhraní, což je velmi nepohodlné řešení. Proto (původně na platformě Microsoft Windows) vzniklo univerzální aplikační rozhraní (API) *ODBC*. Jeho obdobou je *JDBC*, které je charakteristické pro databázi Oracle díky široké podpoře Javy (viz. kapitola 2.1.4).

### ODBC

[2] *ODBC* (Open DataBase Connectivity) je dnes nejrozšířenějším rozhraním pro přístup k databázím a je také téměř všemi databázemi podporováno. Slouží jako prostředník mezi klientskou aplikací a databázovým serverem. Rozhraní *ODBC* se volá jednotně a *ODBC*-ovladač pak požadavek předá databázovému serveru pomocí správného protokolu.

Počáteční nevýhodou byl nižší výkon oproti nativním ovladačům. Staré *ODBC* ovladače sloužily pouze jako mezistupeň mezi aplikací a nativním protokolem databáze. Novější *ODBC* ovladače však přistupují k databázovému serveru přímo a jejich výkon je srovnatelný s nativními ovladači. Některé databázové systémy používají *ODBC* jako svůj jediný nativní protokol.



## JDBC

[9, 21] *JDBC* (Java DataBase Connectivity) je standardizované rozhraní pro přístup k relačním databázím z programů napsaných v Javě. Ve své podstatě je podobné rozhraní ODBC, ale v javových programech není z důvodu bezpečnosti a přenositelnosti programu žádoucí volat kód v jazyce C (ve kterém je napsáno ODBC).

Práce s databází prostřednictvím JDBC se dá charakterizovat následující posloupností akcí [9]:

1. Založení spojení s databází
2. Zasílání příkazů SQL
3. Zpracování výsledků příkazů SQL
4. Ukončení spojení s databází

Pro vlastní implementaci přístupu k databázím slouží tzv. *ovladače JDBC*, které provádějí veškerou komunikaci s cílovou databází. Vzhledem k tomu, že Java je silně objektově orientovaný jazyk, probíhá vše prostřednictvím objektů. Objektem je jak příkaz SQL předávaný databázi, tak výsledek příkazu. Jednotlivé řádky výsledku příkazu SQL jsou přístupné sekvenčně – po spuštění dotazu SQL je přístupný první řádek výsledku a po zavolání příslušné metody se zpřístupňují další řádky.

## 5. Návrh a vytvoření webové databázové aplikace

Cílem této části mé diplomové práce je navrhnout a vytvořit databázovou aplikaci pro správu výsledků úloh studentů. Aplikace musí umožňovat studentům prostřednictvím webového prohlížeče procházet výsledky úloh a zároveň správci stejnou cestou výsledky spravovat. Její pracovní název je *SUP* (Správce Úloh a Předmětů).

Aplikace bude využívat funkce pro přístup k databázi pomocí standardizovaného rozhraní ODBC (viz. kapitola 4.4) a v kombinaci se skriptovacím jazykem PHP (dále kapitola 5.1.1) tak bude možný přenos na jinou databázovou platformu i jiný operační systém (např. Linux).

### 5.1 Volba software

K tomu, aby webová stránka zobrazila konkrétní informaci, musí požadavek od uživatele vyvolat spuštění programu (*skriptu*), který si informace „obstará“ (nejčastěji z databáze) a vygeneruje výsledný HTML dokument.

Při volbě skriptovacího jazyka máme na výběr ze dvou typů [36]:

1. **Serverem vkládané vsuvky** – příkazy skriptu se kombinují přímo s HTML kódem. Předtím než je stránka odeslána uživateli, jsou všechny příkazy skriptu vyhodnoceny WWW serverem a jejich výsledek doplněn na odpovídající místo stránky. Typickými zástupci jsou *SSI* (*Server Side Includes*), *LiveWire* (Netscape), *ASP* (*Active Server Pages*) od Microsoftu a *PHP* (*Personal Home Pages*) – tento systém jsem použil při tvorbě aplikace (viz. dále).
2. **CGI skripty** – samostatné programy napsané v libovolném programovacím jazyce. Pokud server obdrží požadavek URL, které ukazuje na program, spustí ho a výsledek (ve formátu HTML) předá prohlížeči jako odpověď.

[34, 36] Při výběru databáze jsem (samozřejmě) vybíral z volně šiřitelných systémů. Nejznámějšími zástupci jsou *MySQL* a *PostgreSQL*. Oba mají své přednosti i nedostatky. *PostgreSQL* je výhodné podporou transakčního zpracování dat (viz. závěr kapitoly 3.2), uložených procedur<sup>29</sup> a triggerů<sup>30</sup> nebo možností vyššího

---

<sup>29</sup> Procedura je databázový program. Spustí se voláním z jiného programu či procedury, funkce, triggerem nebo ručně.

<sup>30</sup> Trigger je typ uložené procedury, která se spouští při určité akci s konkrétní tabulkou (např. při vládání nebo mazání záznamu) [1].

zatížení. Protože jsem (jako většina uživatelů) volil MySQL hovoří jeho vyšší výkon<sup>31</sup>, podpora dlouhých textových polí a velmi dobrá spolupráce s PHP.

### 5.1.1 PHP

PHP (Personal Home Pages) je skriptovací jazyk, který se přímo začleňuje do HTML kódu. Jeho hlavní předností je možnost bezplatného šíření a podpora více platforem – aplikaci tak můžeme vytvořit ve Windows a poté ji umístit na unixový server.

Důležitou vlastností PHP jsou kromě podpory pro práci s databázemi pomocí standardizovaného rozhraní ODBC i nativní funkce pro práci s většinou databázových systémů (např.: Informix, MS SQL Server, MySQL, Oracle, PostgreSQL, Sybase, ...) <sup>32</sup>. Kromě toho obsahuje i funkce pro práci se staršími databázemi jako dBase.

Příkazy PHP se od ostatního HTML kódu oddělují znaky `<? a ?>`.

Popis instalace a konfigurace PHP na platformě Windows je uveden v Dodatku C.

### 5.1.2 MySQL

[33, 2] MySQL je výkonný a stabilní databázový systém typu klient/server. Podporuje širokou škálu platforem a pro nekomerční účely je šířen zadarmo. Disponuje širokou škálou datových typů, přičemž dokáže uchovat až 4 GB dat v jedné tabulce. Další podstatnou vlastností je i plná podpora české národní abecedy.

MySQL zatím nepodporuje transakční zpracování dat, uložené procedury, triggerů a vnořené dotazy, avšak tyto nedostatky je možno v případě potřeby odstranit vhodně napsanými skripty.

Některé testy sice hovoří o limitu 15 stažených stránek za 1 sekundu, ale většina webových aplikací na Internetu (včetně té naší) nedosahuje ani zdaleka zmíněného limitu [33]. Pokud předpokládáme vyšší zatížení, musíme volit některý z komerčních produktů – například Oracle.

Popis instalace a konfigurace MySQL na platformě Windows je uveden v Dodatku C.

### 5.1.3 Další nástroje

Aby bylo možno skripty používat, musíme je umístit na web server, který je podporuje. V současné době je nejrozšířenějším WWW serverem **Apache**. Je to

---

<sup>31</sup> Na vygenerování jedné stránky potřebuje 2 – 3 krát kratší čas než PostgreSQL [33].

<sup>32</sup> Přehled a popis všech funkcí PHP pro práci s různými databázovými systémy je možno nalézt na <http://download.php.net/manual/en/>

výkonný a stabilní server původně určený pro platformu Unix. Podporuje mnoho moderních technologií a navíc je k dispozici zdarma. Server Apache jsem použil při testování PHP skriptů na vlastním počítači. Popis instalace a konfigurace serveru Apache na platformě Windows je uveden v Dodatku C.

Pro kontrolu funkčnosti skriptů a návrh datové základny je výhodné použít grafického správce databáze MySQL. Takových programů je na internetu<sup>33</sup> k dispozici celá řada. Pro tyto účely jsem použil program **DBTools**<sup>34</sup>. Poskytuje mimo jiné grafické rozhraní pro vzdálenou i „místní“ správu jednotlivých databází, tabulek a uživatelů. Další možností je import a export dat nebo *SQL Query Editor*, který jsem využil pro testování SQL dotazů následně použitých v PHP skriptech.

Samotné PHP skripty je možno psát v libovolném textovém editoru jako *Notepad*, uzpůsobených HTML editorech jako je *HomeSite*<sup>35</sup> či volně šiřitelný *1st Page*<sup>36</sup> nebo speciálních editorech pro programování. Použil jsem editor **PHPEd**<sup>37</sup>. Jde o volně šiřitelný nástroj, který kromě klasických editačních funkcí, knihoven příkazů nebo číslovaných řádků, zobrazuje zdrojový kód přehledně barevně odlišený.

## 5.2 Návrh datové základny

Datová základna je navržena tak, aby při libovolném počtu vložených předmětů, studentů, testů nebo výsledků zůstal počet tabulek i sloupců konstantní. Celkem je rozdělena do pěti tabulek (se sloupci):

1. **Predmety** – pro uložení údajů o předmětech
  - *id\_pred* : identifikační číslo předmětu (int, primární klíč)
  - *nazev* : název předmětu (varchar(40))
2. **Studenti** – pro uložení údajů o studentech
  - *id\_stud* : identifikační číslo studenta (int, primární klíč)
  - *jmeno* : jméno studenta (varchar(15))
  - *prijmeni* : příjmení studenta (varchar(20))
  - *kruh* : příslušnost ke studijnímu kruhu (int)
  - *email* : kontaktní email studenta (varchar(40))

---

<sup>33</sup> Celá řada nástrojů a ovladačů pro databázi MySQL a platformu Windows je k dispozici např. na <http://www.mysql.org/downloads/os-win32.html>

<sup>34</sup> <http://www.dbtools.com.br>

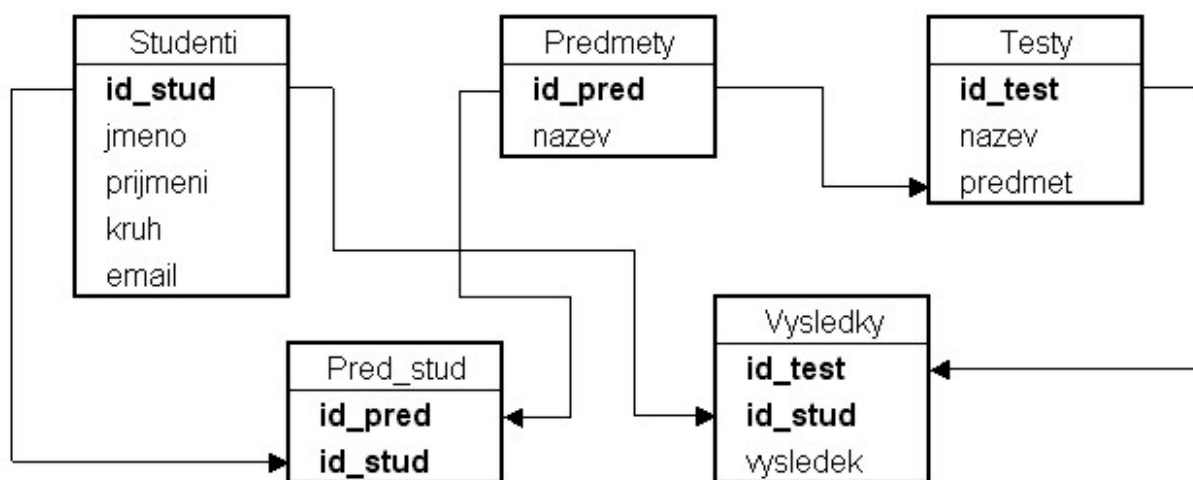
<sup>35</sup> <http://www.allaire.com>

<sup>36</sup> <http://www.evrsoft.com/1stpage>

<sup>37</sup> <http://www.soysal.com/PHPEd>

3. **Pred\_stud** – pro určení příslušnosti studentů k předmětu – každý student má tolik záznamů v kolika je zapsán předmětech (*id\_pred* a *id\_stud* tvoří dohromady *složený primární klíč*)
  - *id\_pred* : identifikační číslo předmětu (int, primární klíč)
  - *id\_stud* : identifikační číslo studenta (int, primární klíč)
4. **Testy** – pro uložení údajů o testech
  - *id\_test* : identifikační číslo testu (int, primární klíč)
  - *nazev* : název testu (varchar(40))
  - *predmet* : příslušnost testu k předmětu (int)
5. **Vysledky** – pro uložení jednotlivých výsledků úloh (*id\_test* a *id\_stud* tvoří dohromady *složený primární klíč*)
  - *id\_test* : identifikační číslo testu (int, primární klíč)
  - *id\_stud* : identifikační číslo studenta (int, primární klíč)
  - *vysledek* : výsledek úlohy (int)

SQL příkazy pro vytvoření všech tabulek jsou uvedeny v příloze 9.2.



Obr. 5.2: Schéma datové základny aplikace



### 5.3.3 Poznámky ke stavbě skriptů

#### **HTTP hlavička:**

Všechny skripty začínají řádkem, který pomocí HTTP hlavičky *Expires* odešle čas vypršení platnosti stránky, který je nastaven na aktuální čas. Tím docílíme, že stránka bude pokaždé načítána ze serveru (ne z vyrovnávací paměti) a tudíž v aktuálním stavu.

```
Header("Expires: ".GMDate("D, d M Y H:i:s")." GMT");
```

#### **Ošetření chyb:**

Před příkazy, které nemusí být vždy provedeny úspěšně, je výhodné použít “zavináč” (@). Tím se potlačí vypsání chybového hlášení PHP, které by mohlo uživatele vylekat. Případnou chybu ošetříme srozumitelnějším způsobem – například:

```
do
{
    @$spojeni = ODBC_Connect("cviceni", "student", "student");
    if (!$spojeni):
        echo "Spojení s databází se nepodařilo navázat.\n";
        break;
    endif;
    . . . . .
} while (false);
ODBC_Close($spojeni);
```

Pokud proměnná *\$spojeni* (popř. jiná z funkcí těla skriptu) vrátí hodnotu typu *false* vypíše se chybové hlášení a příkaz *break* ukončí běh cyklu *do – while*. Za tento cyklus umístíme části kódu, které jsou třeba vykonat vždy – např. uzavření spojení s databází nebo ukončení HTML stránky.

#### **Formátování HTML stránek:**

Pro zjednodušení formátování textu, nadpisů a vlastností stránek bylo použito *kaskádových stylů CSS* – viz. příloha 9.2 .

```
<link rel=stylesheet type="text/css" href="style.css">
```

#### **Generování identifikačních čísel:**

V tabulkách *predmety*, *studenti* a *testy* tvoří primární klíč identifikační číslo, které musí být v rámci jedné tabulky unikátní. Z tohoto důvodu necháme při vkládání nového záznamu generovat ID jako nejvyšší hodnotu z již existujících hodnot. Například pro vložení záznamu do tabulky *studenti* (část kódu z *insert\_stud.php*):

```

$dotaz_na_id = "SELECT Max(id_stud)+1 FROM studenti";
@$vysledek = ODBC_Exec($spojeni, $dotaz_na_id);
  if (!$vysledek):
    echo "Chyba, nové ID nezjištěno.\n";
    break;
  endif;
if (ODBC_Fetch_Row($vysledek))           //přečtení nového ID
  $id_stud = ODBC_Result($vysledek, 1);
else {
  echo "Chyba, nové ID nezjištěno.";
  break;
}
if ($id_stud=="")                       // je-li záznam první,
  $id_stud = 1;                           // id_stud=1

```

### **Zachování integrity dat**

Integritou se rozumí soulad dat se skutečným stavem. Například pokud vymažeme záznam z tabulky *studenti*, musí být odstraněny i odpovídající záznamy z tabulek přiřazení studentů do předmětů (*pred\_stud*) a výsledků (*vysledky*), aby v databázi nezůstala nepotřebná data. O zachování integrity dat se obecně starají uložené procedury a triggery přímo v databázi. Jinou možností (v případě MySQL jedinou – viz. kapitola 5.1.2) je samotný skript. Ten musí být napsán tak, aby postupně (v tomto případě smazal) všechny související záznamy. Příkladem jsou skripty *delete\_\*.php* v příloze 9.2.

## **5.4 Struktura a funkce aplikace**

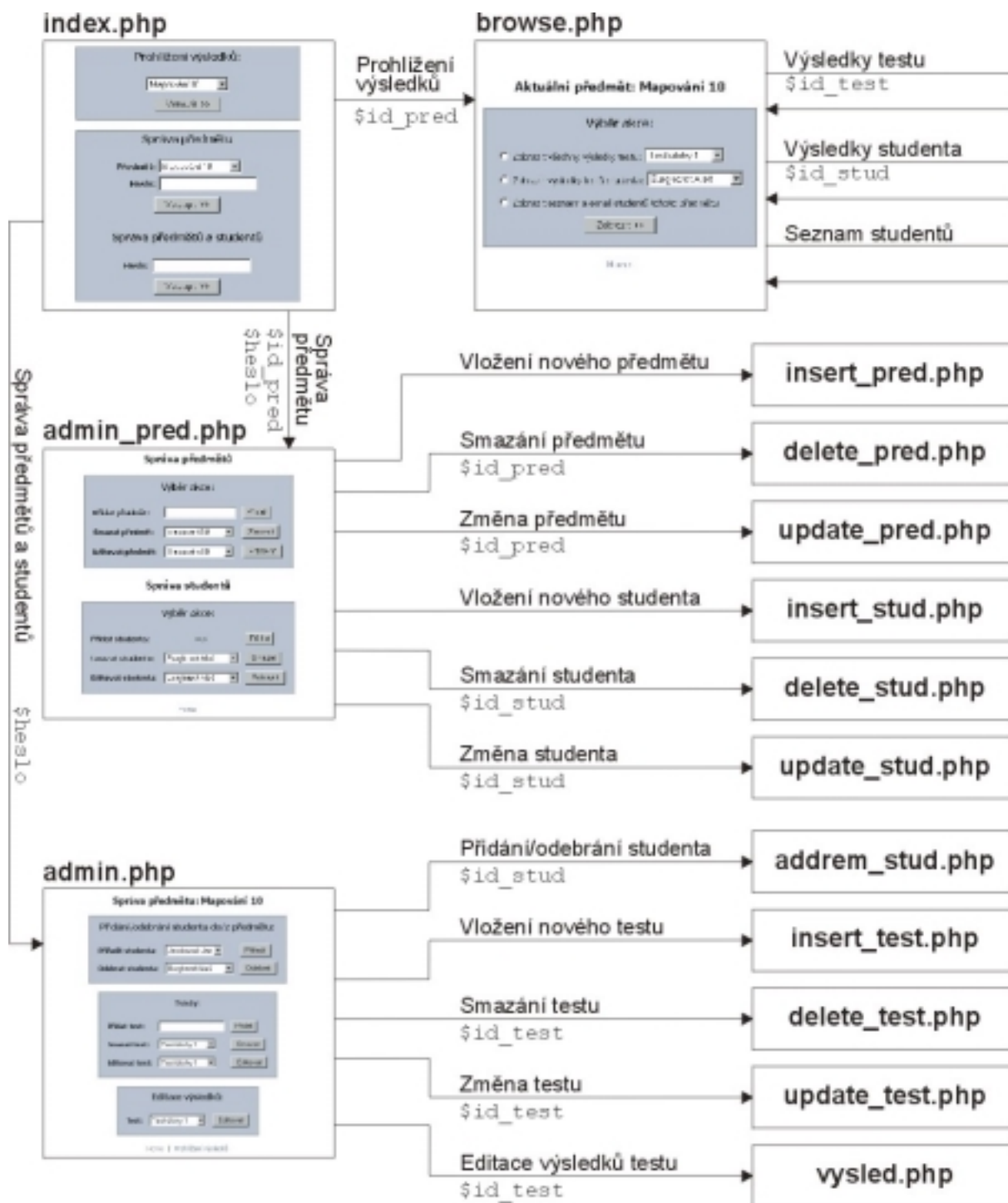
Nyní zde bude popsána struktura a rozložení funkcí vytvořené aplikace (viz. obrázek 5.4 -1), včetně jmen jednotlivých skriptů a hlavních předávaných parametrů, které odpovídají názvům sloupců v příslušných tabulkách (viz. kapitola 5.2). Všechny zdrojové kódy – viz. příloha 9.3 .

### **Úvodní strana**

Úvodní strana (*index.php*) větví aplikaci na tři základní části:

1. **Prohlížení výsledků** – poskytuje veřejný přístup k prohlížení všech výsledků vybraného předmětu (parametr *id\_pred*).
2. **Správa předmětu** – po zadání hesla umožňuje spravovat testy a jejich výsledky v daném předmětu (parametr *id\_pred*), včetně přiřazování studentů.
3. **Správa předmětů a studentů** – po zadání příslušného hesla dovoluje spravovat všechny předměty a studenty.





Obr. 5.4 – 1: Schéma aplikace

## Prohlížení výsledků

Nabízí volný přístup k prohlížení výsledků (*browse.php*) konkrétního testu (patřícího ke zvolenému předmětu) nebo všech výsledků daného studenta (zapsaného ve zvoleném předmětu). Jako doplněk umožňuje zobrazit seznam s kontakty studentů aktuálně zapsaných v předmětu.

## Seznam studentů v aktuálním předmětu

### Výsledky: Test úlohy 2

Burghardt Aleš	1
Doležalová Jana	2
Eisnerová Kristýna	2
Mencl Robert	3
Plavec Jan	2
Vejnar Marek	2
Švejda Jiří	2

ID	Příjmení a jméno	Kruh	E-mail
1	Burghardt Aleš	62	burgy@email.cz
9	Doležalová Jana	64	jana.dolezalova@email.cz
6	Eisnerová Kristýna	64	kristina.eisnerova@email.cz
7	Mencl Robert	65	robac@centrum.cz
4	Plavec Jan	64	plavcik@sin.cvut.cz
5	Vejnar Marek	62	ignacius@email.cz
3	Švejda Jiří	62	jiri.svejda@email.cz

Obr. 5.4 – 2, 3: Příklady výstupu skriptu *browse.php*

### Správa předmětu

Na této stránce (*admin.php*) jsou k dispozici tři základní funkce:

1. **Přidání/odebrání studenta z/do předmětu** – umožňuje přidávat (resp. odebírat) studenty do předmětu, přičemž u přidávání výběr nabízí jen ty, kteří ještě v předmětu nejsou a u odebrání naopak. Obslužnému skriptu (*addrem\_stud.php*) je předáván parametr *id\_stud*.

### Editace výsledků: Závěrečný test

Burghardt Aleš	<input type="text" value="1"/>
Doležalová Jana	<input type="text" value="2"/>
Eisnerová Kristýna	<input type="text" value="1"/>
Mencl Robert	<input type="text" value="2"/>
Plavec Jan	<input type="text" value="1"/>
Vejnar Marek	<input type="text" value="2"/>
Švejda Jiří	<input type="text" value="1"/>


Obr. 5.4 – 4: Editace výsledků testu – *vysled.php*

2. **Testy** – dovoluje vytvářet (*insert\_test.php*), mazat (*delete\_test.php*) a editovat (*update\_test.php*) testy v rámci aktuálního předmětu. Skriptům je předáván parametr *id\_test*.
3. **Editace výsledků** – slouží k souhrnné editaci výsledků vybraného testu. Skriptu je předán parametr *id\_test*.

### Správa předmětů a studentů

Tato sekce (*admin\_pred.php*) obsahuje nástroje pro správu:

1. **Předmětů** – pomocí předávaného parametru *id\_pred* umožňuje předměty vytvářet (*insert\_pred.php*), mazat (*delete\_pred.php*) a editovat (*update\_pred.php*).
2. **Studentů** – pomocí předávaného parametru *id\_stud* umožňuje studenty vytvářet (*insert\_stud.php*), mazat (*delete\_stud.php*) a editovat (*update\_stud.php*).



The screenshot shows a web form titled "Editace studenta" with a light blue background. It contains the following fields and a button:

- ID:** 1
- Jméno:**
- Příjmení:**
- Kruh:**
- E-mail:**
- Změnit** button

Obr. 5.4 – 5: Editace studenta – *update\_stud.php*

## 5.5 Budoucí vývoj aplikace

V případě uvedení do praktického provozu bude účelné upravit, případně přidat, některé funkce ke zvýšení bezpečnosti a praktičnosti aplikace.

### Prohlížení výsledků

Pro přehlednost bude výhodnější zobrazovat výsledky všech testů daného předmětu najednou, na rozdíl od stávajícího prohlížení výsledků testu nebo studenta.

## Hesla

V této první verzi aplikace pracuje s jedním konstantním heslem „napevno“ uvedeným ve skriptech. Bude účelné oddělit heslo pro správu předmětů a studentů od správy konkrétního předmětu. Každý předmět bude mít své vlastní heslo uložené přímo v databázi (nový sloupec v tabulce *predmety*). K zabezpečení hesel uložených v databázi nabízí MySQL funkci *PASSWORD()*, která heslo před uložením zakóduje a dále již pracuje jen s jeho zakódovanou formou.

S tím souvisí i přidání skriptu, který umožní hesla vkládat a měnit.

## Email

Další praktickou funkcí může být odesílání emailu z webového formuláře pro všechny zapsané (popřípadě vybrané) studenty. Pro tento účel disponuje PHP funkcí *Mail()* – viz. příklad:

```
if (@Mail("email_adresáta/adresátů", "subject",
          "text_zpravy"[,"hlavičky_emailu"]))
    echo "Email úspěšně odeslán.\n";
else
    echo "Email se nepodařilo odeslat.\n";
```

## 6. Závěr

Produkty firmy Oracle se v posledních letech staly synonymem pro internetové technologie. Tato práce se zabývala možnostmi databázového systému Oracle8i prezentovat a spravovat data uložené v databázi prostřednictvím World Wide Webu. Kromě stručných teoretických základů a popisu hlavních součástí Oracle8i jsou stěžejní tři kapitoly – *Oracle Web Publishing Assistant* (kapitola 4.2), *Oracle WebDB* (kapitola 4.3), návrh a realizace vlastní databázové aplikace (kapitola 5).

### Shrnutí a porovnání zmíněných metod:

1. **Oracle Web Publishing Assistant** je standardní součástí Oracle8i. Dokáže vytvářet výstupy ve formátu HTML na základě databázových tabulek nebo konkrétního SQL dotazu. Má široké možnosti nastavení obnovování (znovuvytvoření) www stránky – tzn., že při nastavení obnovování v řádu minut nebo při změně dat v databázi (s určitým omezením) se stránky prakticky blíží dynamickým (z hlediska aktuálnosti zobrazovaných dat). Práce s ním je velice jednoduchá a rychlá, avšak hodí se pouze pro prezentaci dat z tabulek aktualizovaných vnitřními nástroji databáze.
2. **Oracle WebDB** je samostatně dodávaná komponenta, která prostřednictvím webového prohlížeče umožňuje jak kompletní správu databáze (tabulek, uživatelů apod.), tak i tvorbu, správu a monitorování webových aplikací uložených přímo v databázi. Bez nutné znalosti programování, SQL nebo HTML dovoluje pomocí průvodců poměrně snadno a rychle vytvářet jednotlivé komponenty a spojovat je v ucelené webové aplikace – prohlížení dat v podobě HTML tabulek a přístup k nim prostřednictvím navržených formulářů. Nutno ovšem dodat, že bez naprosté znalosti SQL nebo principů web-aplikací by byla tvorba obtížná – vzhledem k velkému množství parametrů, kterými lze naopak aplikaci přizpůsobit našim představám.
3. **Vlastní aplikace** je samozřejmě nejuniverzálnější řešení. Můžeme ji přesně přizpůsobit daným požadavkům – jak z hlediska funkčnosti, tak i po vizuální stránce. Nezanedbatelnou výhodou je též transparentnost (alespoň pro autora) procesů, které aplikace v různých situacích provádí – např. při změnách dat v databázi nebo údržbě jejich integrity. To usnadňuje případné pozdější úpravy nebo přenos celé aplikace. Veškerý potřebný software pro

vytvoření i provoz je dnes možné volit t kategorie volně šiřitelných produktů a tím minimalizovat celkové náklady.

Neopomenutelnou podmínkou pro realizaci je však orientace v potřebných informačních technologiích. Jako např. detailní znalost některého programovacího jazyka pro celkovou stavbu, dotazovacího jazyka SQL pro komunikaci s databází či HTML pro formátování výstupu skriptů a zpřístupnění ve webovém prohlížeči.

Aplikace podobného typu jsou vhodné zejména pro menší či střední projekty nebo menší firmy, kde bude dostačující rychlost, výkon a úroveň zabezpečení. Naproti tomu komerční produkty Oracle nabízí mimo jiné vysoký výkon a bezpečnost, množství dokumentace, technickou podporu mnoho kvalitních nástrojů pro správu databáze. Z toho plyne i oblast jejich nasazení. Oracle v ČR například používají firmy Eurotel, Český Telecom, Škoda Auto, Seznam.cz či Komerční banka nebo ve světě internetoví giganti jako Yahoo.

Je zřejmé, že výběr databáze či jednotlivých nástrojů vždy souvisí s oblastí nasazení a ostatními individuálními kritérii jako výkon, kapacitní omezení, podporované platformy, počet současně připojených uživatelů nebo bezpečnostní požadavky.

Celá diplomová práce včetně příloh bude v HTML a PDF verzi zpřístupněna na Internetu na adrese <http://www.antinova.cz/hrustic/dipl> a funkční databázová aplikace popsaná v kapitole 5 na adrese <http://geo.fsv.cvut.cz/sup/index.php>.

## 7. Dodatky

### A. Instalace Oracle8i Enterprise Edition 8.1.6

Databázový systém Oracle8i je k dispozici pro celou řadu platform. Jako první jsem měl k dispozici verzi pro Linux (verzi 8.0.5 a poté 8.1.7), která se mi však i po dlouhém úsilí nepodařila nainstalovat. Příkladal jsem to svým slabým znalostem linuxu, avšak zkušenější kolegové objevili i některé nekompatibility s linuxovými distribucemi (konkrétně RedHat 7.0).

Databáze Oracle jsou poměrně těsně integrovány s prostředím operačního systému Windows NT. Díky firmě Oracle Czech s.r.o. jsem měl k dispozici Oracle8i Enterprise Edition for Windows NT Release 2 (8.1.6). Instalace této verze již proběhla bez problémů a bude zde nyní stručně popsána (podrobnější informace – viz. [15]).

#### Minimální systémové požadavky:

Operační systém	Windows NT 4.0 a Service Pack 5.0+
Procesor	Pentium 166 (doporučeno Pentium 233)
RAM	96 MB (doporučeno 256 MB)
Hard disk	cca 1 GB

#### Instalace

Od verze 8.1 je k dispozici *Oracle Universal Installer*, který je napsán v Javě a tudíž nezávislý na platformě.

1. V úvodním *autorun* okně zvolíme *Install/Deinstall Products*
2. ⇒ *Next*
3. Source ponecháme a nastavíme **Oracle Home Name** např. `OraHome81` a jako **Oracle Home Path** bude nabídnut disk s největším místem a cestou – např.: `D:\Oracle\Ora81`  
⇒ *Next*  
Část `D:\Oracle` se nazývá **Oracle Base** (v proměnné `ORACLE_BASE`)
4. Zvolíme *Oracle8i Enterprise Edition 8.1.6.0.0* a dále ⇒ *Next*
5. Zvolíme *Typical* a dále ⇒ *Next*

6. Systém (volitelně) vytvoří úvodní databázi, kde zadáme (viz. popis v kap. 3.2)  
*Global Database Name:* dbatest.domain  
*SID:* dbatest
7. V *Summary* okně zvolíme *Install*
8. Po proběhnutí instalace potvrdíme hlášení o vytvoření databáze ⇒ *OK*
9. V okně *End of Installation* ukončíme instalaci *Exit*

Všechny nainstalované produkty jsou nyní dostupné v programových skupinách „Oracle - Oracle\_Home\_Name“ a „Oracle Installation Products“.

## B. Instalace Oracle WebDB 2.2

WebDB je dodáván na samostatném CDROM a opět jsem ho měl k dispozici od Oracle Czech s.r.o.. Bude popsána typická instalace. Parametr *Oracle Home* musí být jiný než u Oracle8i. Více informací – viz. originální dokumentace.

### Instalace

1. Na CDROM spustíme *WT\setup.exe*
2. *Company Name:* libovolné  
*Oracle Home Name:* WebDB  
*Location:* D:\Oracle\WebDB  
*Language:* English (implicitní hodnota) ⇒ *OK*
3. Zvolíme *Typical Install* ⇒ *OK*
4. Zvolíme *Oracle8i* ⇒ *OK*
5. V okně *Connect to the database* zadáme:  
*Password:* heslo uživatele SYS – implicitně: change\_on\_install  
*TNS Names Alias:* zpravidla je nutno vytvořit nový - zadáme nový alias (např. webdb) a vytvořením nás automaticky provede *Net8 Easy Config*  
*Host Name:* jméno počítače – viz. Start⇒Settings⇒Control panel⇒Network  
*WebDB Listener Port #:* 80
6. Informační okno ⇒ *OK*
7. *Warning* okno ⇒ *YES*
8. V okně *Net8 Easy Config* zvolíme *Add New Service (webdb)* ⇒ *Next*
9. Zvolíme *TCP/IP (Internet Protocol)* ⇒ *Next*



10. *Host Name*: viz. bod 5  
*Port Number*: ponecháme 1521
11. *Database SID*: dbatest (viz. kapitola 3.3.1) ⇒ *Next*
12. ⇒ *Test Service*
13. Zadáme *Username/Password* – např. *system/manager* ⇒ *Test*
14. Pokud test proběhne v pořádku ⇒ *Done*
15. ⇒ *Finish*
16. Informační okno o vytvoření *TNS Names Alias* ⇒ *OK*
17. *Installation Schema*: WEBDB  
Položky *Tablespace* můžeme ponechat ⇒ *OK*
18. Informační okno o jméně a heslo pro vstup do WebDB ⇒ *OK*
19. Okno *Choose Languages* (v tomto případě bez češtiny) ⇒ *OK*
20. Informační okno o startu instalace WebDB ⇒ *YES*
21. Informační okno jak vstupovat do WebDB ⇒ *OK*

Standardně se nainstaluje a nakonfiguruje i webový server *Oracle WebDb Listener* (pod Windows NT jako služba – viz. *Start* ⇒ *Settings* ⇒ *Control panel* ⇒ *Services*), což deaktivuje případný stávající webový server (např. *Apache*). Pro aktivaci původního je nutné WebDB Listener zastavit (*Services* ⇒ *Stop*).

## C. Instalace a konfigurace Apache, PHP a MySQL

V pozadí mnoha dnešních dynamicky generovaných webových stránek stojí kombinace webserveru *Apache*, scriptovací jazyk *PHP* (Personal Home Pages) a databázový server *MySQL*. V originální dokumentaci i na webových stránkách lze nalézt množství informací o instalaci a konfiguraci jednotlivých produktů, avšak často jsou až příliš obsáhlé. Proto zde budou uvedeny jen potřebné kroky pro instalaci na platformě Windows.

### Apache

1. Stáhneme si instalační soubor pro příslušnou platformu např. z ***www.apache.org*** (v tomto případě *apache\_1\_3\_11\_win32.exe*)
2. Soubor (spuštěním) nainstalujeme např. do adresáře *C:\apache*. Konfigurační soubory jsou v adresáři *C:\apache\conf*.

3. V souboru **httpd.conf** nastavíme položku **SERVERNAME** (jméno serveru) např. na *localhost*. A dále položku **DOCUMENTROOT** (adresář, kde budou uloženy dokumenty – obsah serveru) např. na “C:\3w”.
4. Server spustíme příkazem *Start Apache* a zastavíme příkazem *Stop Apache*, které se standardně vytvoří v programové skupině *Apache Web Server*.

Ve zmíněné programové skupině lze použít i příkaz *Install Apache as a service*, který (pod Win NT) nastaví Apache jako službu spouštěnou automaticky při startu počítače.

## PHP

1. Stáhneme si instalační soubor pro příslušnou platformu z **www.php.net** (v tomto případě *php-3.0.15-win32.zip*)
2. Soubor (ZIP) rozbalíme např. do C:\php3
3. Všechny soubory s příponou **DLL** zkopírujeme do hlavního adresáře Windows (pro Win NT/2000/XP je to C:\winnt\system32 a pro Win 95/98/Me C:\windows\system)
4. V adresáři C:\php3 přejmenujeme soubor *php3.ini-dist* na **php3.ini** a přesuneme ho do C:\winnt resp. do C:\Windows.
5. V souboru C:\apache\conf\httpd.conf najdeme řádky začínající „ScriptAlias“ a přidáme další: **ScriptAlias /php3/ “C:/php3“**.

Dále najdeme řádek **AddType application/x-httpd-php3 .phtml** a „odkomentujeme“ ho – tj. smažeme křížek (#) na začátku řádku. Na jeho konec ještě kromě *.phtml* přidáme další přípony souborů, které budou obsahovat PHP kód. Obvykle to jsou: *\*.php* nebo *\*.php3* (přidáme **.php .php3**).

Nakonec do *httpd.conf* přidáme dva řádky **AddHandler phpskript .php** a **Action phpskript /php3/php.exe**, který říká, kde se mají soubory s PHP kódem vyhodnotit.

## MySQL

1. Stáhneme si příslušný soubor (pro Windows) např. na **www.mysql.org** (zde konkrétně *mysql-shareware-3.22.34-win.zip*)
2. Soubor rozbalíme a spustíme instalátor. MySQL nainstalujeme do C:\mysql

3. V souboru `C:\php3\php3.ini` najdeme řádek **`extension=php3_mysql.dll`** a “odkomentujeme” ho – tj. smažeme středník (;) na začátku řádku.  
Najdeme řádek s **`extension_dir`** a nastavíme ho na adresář, ve kterém máme knihovny PHP (*dll*). Pro Windows NT to je `C:\winnt\system32` (viz. předchozí kapitola)
4. MySQL server spustíme souborem `C:\mysql\bin\mysqld-shareware.exe`

## Test funkčnosti

Funkčnost všech produktů dohromady můžeme otestovat následovně:

Vytvoříme soubor např. `C:\3w\test.php` a do něj vložíme kód:

```
<?
PHPInfo();
?>
```

Spustíme webový prohlížeč a do adresy zadáme *localhost* (popř. IP adresu 127.0.0.1). Měl by se zobrazit obsah adresáře 3w a v něm zvolíme soubor *test.php*. Je-li vše v pořádku, objeví se tabulka s informacemi o aktuální verzi PHP, která musí obsahovat řádek nazvaný *mysql* a informace o něm.

## D. Konfigurace ODBC

Abychom k databázi mohli přistupovat pomocí universálních ODBC funkcí (viz. kapitola 4.4 a 5.3.2), musíme nastavit příslušný ODBC zdroj (*DSN – Data Source Name*). Každý ODBC zdroj odpovídá jedné konkrétní databázi (u Oracle *dbatest* a u MySQL *cviceni*), a proto je nutné provést nastavení pro každou databázi resp. ODBC zdroj zvlášť.

### MySQL

1. Stáhneme si ODBC ovladače např. z **[www.mysql.org](http://www.mysql.org)** pro příslušnou platformu (příklad souboru pro Win NT: *myodbc-2.50.37-nt.zip* a pro Win 9x: *myodbc-2.50.37-win95.zip*).
2. Soubor rozbalíme a spustíme *Setup.exe*.
3. Potvrdíme úvodní hlášení a vybereme ze seznamu *MySQL* a zvolíme *Setup*. (popřípadě instalaci ukončíme *Close* a DSN nastavíme později – viz. *Settings* ⇒ *Control Panel* ⇒ *ODBC Sources* ⇒ v záložce *System DSN* vybereme *MySQL* a zvolíme *Add*).

4. V okně *TDX mysql Driver default configuration* nastavíme parametry pro připojení k databázi *cviceni* na lokálním počítači:  
*Windows DSN name* – jméno datového zdroje – je užitečné volit stejné jako Database Name (cviceni)  
*MySQL host (name or IP)* – doména počítače s databází (localhost)  
*MySQL Database Name* – jméno databáze (cviceni)  
Popřípadě zadáme uživatelské jméno a heslo (*User, Password*)
5. Potvrdíme nastavení ⇒ OK
6. Pro zpřístupnění ODBC funkcí v PHP musíme navíc v souboru **php3.ini** odkomentovat (odstranit středník) řádek **extension=php3\_odbc.dll**

## Oracle

Podobně vytvoříme ODBC zdroj i pro oraclovskou databázi a *DBATEST\_SERVER*

1. *Settings* ⇒ *Control Panel* ⇒ *ODBC Sources* ⇒ v záložce *System DSN* zvolíme *Add*.
2. Vybereme *Oracle ODBC Driver* ⇒ *Finish*
3. Vyplníme vstupní údaje:  
*Data source name* – jméno datového zdroje – je užitečné volit ho pro pořádek stejné jako Service name (dbatest\_server)  
*Description* – libovolný popis  
*Service name* (dbatest\_server)  
*UserID* (helmut)

Nově vytvořený ODBC zdroj můžeme otestovat pomocí nástroje *Oracle ODBC Test* (v programové skupině *Oracle – OraHome81* ⇒ *Network Administration*).

Zvolíme *Connect...*, vybereme jméno vytvořeného zdroje (dbatest\_server) a vyplníme vstupní údaje – viz. kapitola 3.3.1. Je-li spojení navázáno, můžeme zadávat libovolné SQL dotazy na databázi.

## 8. Zdroje informací

### Publikace

- [1] McCullough-Dieter C.: Mistrovství v Oracle8, Computer Press, Praha 1999
- [2] Kosek J.: PHP – tvorba interaktivních internetových aplikací, Grada Publishing, 1999
- [3] Šimůnek M.: SQL kompletní kapesní průvodce, Grada Publishing, 1999
- [4] Lacko L.: Web a databáze, Computer Press, Praha 2001
- [5] Riordan M. R.: Vytváříme relační databázové aplikace, Computer Press, Praha 2000
- [6] Perry J. P.: Java – tvorba dokonalých WWW stránek, Grada Publishing, 1996

### Periodika

- [7] Kocan M.: Příloha: Databáze a vývoj aplikací, Computer 21/00, str. 80 - 82
- [8] Kocan M.: Databázový chaos, Computer 11/98, str. 56 - 57
- [9] Relich M.: Databáze a Java, Computer 11/98, str. 58
- [10] Kocan M.: Rozčeřené DB-hladiny (+ slovníček), Computer 11/00, str. 67, 71
- [11] Beran M.: Platit se bude za řešení, Computer 17/99, str. 67
- [12] Fikker J.: Databáze pro Internet, PC WORLD 9/1999, str. 92
- [13] Fikker J.: SQL server Oracle8 for NT pro náročné, PC WORLD 4/2000
- [14] Fikker J.: Databázové aplikace jednoduše – Oracle WebDB 2.0, PC WORLD 4/2000

### Originální dokumentace v elektronické podobě<sup>38</sup>

- [15] Oracle8i Installation Guide for Windows NT
- [16] Oracle Enterprise Manager Administrator's Guide 2.1
- [17] SQL\*Plus User's Guide and Reference
- [18] Oracle Web Publishing Assistant Getting Started for Windows NT
- [19] Oracle WebDB Tutorial Guide, Release 2.2
- [20] Oracle WebDB Creating and Managing Components
- [21] Oracle8i Java Developer's Documentation

---

<sup>38</sup> Dokumenty [15] až [21] jsou součástí originálního dokumentačního balíku *Oracle8i Server On-Line Documentation 8.1.6* dostupný mimo jiné na <http://docs.oracle.com>

- [22] MySQL Reference Manual  
<http://www.mysql.org>
- [23] Berkeley DB Reference Guide  
<http://www.sleepycat.com>

### **Webové stránky a články**

- [24] Oficiální stránky Oracle Corporation resp. Oracle Czech s.r.o.  
<http://www.oracle.com>
- [25] Unicorn Distribution, Oracle8i Enterprise Edition  
[http://www.unicorngroup.cz/distribution/software/oracle/8i\\_enterprise.htm](http://www.unicorngroup.cz/distribution/software/oracle/8i_enterprise.htm)
- [26] Oracle8i Tutorials, (série článků)  
<http://databases.about.com/oracle>
- [27] Central WebDB Repository, Information about for Oracle WebDB  
<http://www.gizwebs.com/webdb/information.htm>
- [28] Náprstek V.: Oracle WebDB, Linuxové noviny 11/99  
<http://www.linux.cz/noviny/1999-11/clanek07.html>
- [29] Unicorn Distribution, Oracle WebDB  
<http://www.unicorngroup.cz/distribution/software/oracle/webdb.htm>
- [30] Oracle Portal (WebDB) FAQ  
<http://www.orafaq.org/faqwebdb.htm>
- [31] Lim J.: ODBC and DSN – less Connections for PHP  
<http://php.weblogs.com/odbc>
- [32] Lidák P.: Relačný databázový systém typu klient/server I., resp. II.  
<http://programovanie.pc.sk/databazy/clanok.ltc?ID=256>, resp. [ID=265](http://programovanie.pc.sk/databazy/clanok.ltc?ID=265)
- [33] Lidák P.: Databázové možnosti a riešenia – komerčné aj voľne šíriteľné  
<http://programovanie.pc.sk/databazy/clanok.ltc?ID=266>
- [34] Lidák P.: Publikovanie z databáz na internete a intranete  
<http://programovanie.pc.sk/databazy/clanok.ltc?ID=254>
- [35] Apache Server Frequently Asked Questions  
<http://www.apache.org/docs/misc/FAQ.html>
- [36] Kosek J.: Aplikace na Webu, (série článků)  
<http://www.kosek.cz/clanky/iweb/index.html>

## 9. Přílohy

### 9.1 Šablona – Oracle Web Publishing Assistant

```
<html>
<head>
<meta http-equiv="Content-Type"
content="text/html">
<title>Výsledky testů - Web Publishing Assistant</title>
</head>

<body bgcolor="#FFFFFF">
<font color="#FF0000" size="6" face="Arial, Helvetica, sans-serif">
<p align="center"><b>VÝSLEDKY</b></p>
<p></p>
</font>
<table border="1" width="600" bgcolor="#CCCCCC" bordercolor="#000000"
align="center">
  <font face="Arial, Helvetica, sans-serif">
  <tr>
    <td align="center" width="15%"><b>ID</b></td>
    <td align="center" width="25%"><b>Jméno</b></td>
    <td align="center" width="15%"><b>Test 1</b></td>
    <td align="center" width="15%"><b>Test 2</b></td>
    <td align="center" width="15%"><b>Test 3</b></td>
    <td align="center" width="15%"><b>Test 4</b></td>
  </tr> </font>
</table>
<table border="0" width="600" bordercolordark="#000000"
bordercolorlight="#FFFFFF" bordercolor="#000000" align="center">
  <%begindetail%>
  <tr>
    <td width="15%" height="24" align="center"><%id%></td>
    <td width="25%" height="24"><%Jméno%></td>
    <td width="15%" height="24" align="center"><%test1%></td>
    <td width="15%" height="24" align="center"><%test2%></td>
    <td width="15%" height="24" align="center"><%test3%></td>
    <td width="15%" height="24" align="center"><%test4%></td>
  </tr>
  <%enddetail%>
</table>
</body>
</html>
```

### 9.2 Definice kaskádových stylů CSS

```
BODY {
  BACKGROUND: #FFFFFF;
  COLOR: black;
  FONT-FAMILY: Tahoma, Arial CE, Arial
  font-weight : normal;
  font-style : normal;
  font-size : 10pt; }

P {
  COLOR: black;
  FONT-FAMILY: Tahoma, Arial CE, Arial
  font-weight : normal;
  font-style : normal;
  font-size : 10pt; }
```

```

TABLE {
    COLOR: black;
    FONT-FAMILY: Tahoma, Arial CE, Arial
    font-weight : normal;
    font-style : normal;
    font-size : 10pt; }
H1 {
    text-align : center;
    font-size : 18pt;
    font-weight : bold;
    FONT-FAMILY: Tahoma, Arial CE, Arial }
H2 {
    text-align : center;
    font-size : 16pt;
    font-weight : bold;
    FONT-FAMILY: Tahoma, Arial CE, Arial }
H3 {
    text-align : center;
    font-size : 14pt;
    font-weight : bold;
    FONT-FAMILY: Tahoma, Arial CE, Arial }
H4 {
    text-align : center;
    font-size : 12pt;
    font-weight : bold;
    FONT-FAMILY: Tahoma, Arial CE, Arial }
A {
    FONT-FAMILY: Tahoma, Arial CE, Arial
    font-size : 10pt;
    font-weight : normal;
    font-style : normal;
    color : #65739A;
    text-decoration : none; }
A:Visited {
    FONT-FAMILY: Tahoma, Arial CE, Arial
    font-size : 10pt;
    font-weight : normal;
    font-style : normal;
    color : #8A95B3;
    text-decoration : none; }
A:Active {
    FONT-FAMILY: Tahoma, Arial CE, Arial
    font-size : 10pt;
    font-weight : normal;
    font-style : normal;
    color : #65739A;
    text-decoration : none; }
A:Hover {
    FONT-FAMILY: Tahoma, Arial CE, Arial
    font-size : 10pt;
    font-weight : normal;
    font-style : normal;
    color : #65739A;
    text-decoration : underline; }
HR {
    color : #BCC2D3;
    width : 600px; }

```



## 9.3 Zdrojové kódy skriptů vytvořené databázové aplikace

Části kódu, které obsahují jen HTML jsou z důvodu úspory místa zhuštěny (na úkor přehlednosti).

### Skript pro úvodní vytvoření databázových tabulek

```
<h4>Předem musí být vytvořena databáze (ODBC zdroj - DSN) "CVICENI"
</h4><br>
<?
$spojeni = ODBC_connect("cviceni","root","") or die("Spojení s DSN zdrojem
se nepodařilo navázat :(");
// tabulka PREDMETY
$stab_predmety = "CREATE TABLE predmety (id_pred INT NOT NULL PRIMARY KEY,
nazev VARCHAR(40))";
$vysedek = ODBC_Exec($spojeni, $stab_predmety) or die("Chyba tab 1 :(");
echo "<b>Tabulka PREDMETY vytvořena ...</b><br>\n";
// tabulka STUDENTI
$stab_studenti = "CREATE TABLE studenti (id_stud INT NOT NULL PRIMARY KEY,
jmeno VARCHAR(15), prijmeni VARCHAR(20), kruh INT, email VARCHAR(40))";
$vysedek = ODBC_Exec($spojeni, $stab_studenti) or die("Chyba tab 2 :(");
echo "<b>Tabulka STUDENTI vytvořena ...</b><br>\n";
//tabulka PRED_STUD
$stab_pred_stud = "CREATE TABLE pred_stud (id_pred INT NOT NULL, id_stud INT
NOT NULL, PRIMARY KEY (id_pred, id_stud))";
$vysedek = ODBC_Exec($spojeni, $stab_pred_stud) or die("Chyba tab 3 :(");
echo "<b>Tabulka PRED_STUD vytvořena ...</b><br>\n";
// tabulka TESTY
$stab_testy = "CREATE TABLE testy (id_test INT NOT NULL PRIMARY KEY, nazev
VARCHAR(40), predmet INT NOT NULL)";
$vysedek = ODBC_Exec($spojeni, $stab_testy) or die("Chyba tab 4 :(");
echo "<b>Tabulka TESTY vytvořena ...</b><br>\n";
// tabulka VYSLEDKY
$stab_vysledky = "CREATE TABLE vysledky (id_test INT NOT NULL, id_stud INT
NOT NULL, vysledek INT, PRIMARY KEY (id_test, id_stud))";
$vysedek = ODBC_Exec($spojeni, $stab_vysledky) or die("Chyba tab 5 :(");
echo "<b>Tabulka VYSLEDKY vytvořena ...</b><br>\n";
$koniec = ODBC_close($spojeni);
?>
```

### index.php

```
<? Header("Expires: ".GMDDate("D, d M Y H:i:s")." GMT"); ?>
<html> <head>
<title>Home - správa výsledků</title>
<link rel=stylesheet type="text/css" href="style.css">
</head>
<body bgcolor="#FFFFFF">
<table width="750" border="0" cellspacing="0" cellpadding="0">
<!-- celková tabulka -->
<tr><td>
<? require "hlava.php"; ?>
</td></tr>
<tr><td><div align="center">
<!-- ----- Prohlížení výsledků ----- -->
<table width="350" border="1" cellspacing="0" cellpadding="1"
bgcolor="#BCC2D3">
<tr><td height="20"><div align="center">
<?
do
```

```

{
@$spojeni = ODBC_Connect("cviceni", "student", "student");
                                //připojení k databázi (jen pro SELECT)
    if (!$spojeni):
        echo "Spojení s databází nenavázáno :( \n";
        break;
    endif;
$dotaz = "SELECT * FROM predmety ORDER BY nazev";
                                //dotaz na všechny předměty
    @$vysledek = ODBC_Exec($spojeni, $dotaz);
    if (!$vysledek):
        echo "Dotaz se nezdařil :( \n";
        break;
    endif;
?>
<table cellpadding="1">
<tr><td colspan="2"><div align="center">
<h4>Prohlížení výsledků:</h4></div>
    </td></tr>
        <form method="POST" action="browse.php">
            <tr><td colspan="2"><div align="center">
<select name="id_pred" size=1>
<?
    while (ODBC_Fetch_Row($vysledek)):
                                //výpis výběru předmětů
echo "<option value=\"".ODBC_Result($vysledek,
"\"id_pred\".\">\".ODBC_Result($vysledek, \"nazev\").\"</option>\n";
    endwhile;
?>
</select>
    </div></td></tr>
        <tr><td colspan="2" align="center" height="37">
<input type="submit" VALUE="Vstoupit &gt;&gt;";
</td></tr>
</form></table></div>
</td></tr></table><br>
<!-- ----- Vstup pro administraci ----- -->
<table width="350" border="1" cellspacing="0" cellpadding="1"
bgcolor="#BCC2D3">
    <tr><td><div align="center">
        <h4>Správa předmětu</h4>
        <table cellpadding="1">
            <form method="POST" action="admin.php">
                <input type="hidden" name="akce" value="login">
            <tr><td><b>Předmět:</b></td><td>
                <select name="id_pred" size=1>
            <?
$dotaz = "SELECT * FROM predmety ORDER BY nazev";
                                //dotaz na všechny předměty
@$vysledek = ODBC_Exec($spojeni, $dotaz);
if (!$vysledek):
    echo "Dotaz se nezdařil :( \n";
    break;
endif;
while (ODBC_Fetch_Row($vysledek)):
                                //výpis výběru předmětů
    echo "<option value=\"".ODBC_Result($vysledek,
"\"id_pred\".\">\".ODBC_Result($vysledek, \"nazev\").\"</option>\n";
    endwhile;
?>
</select></td></tr>

```

```

<tr><td align="right"><b>Heslo:</b></td>
<td><input type="password" name="heslo"></td></tr>
<tr><td colspan="2" align="center" height="37">
<input type="submit" VALUE="Vstoupit &gt;&gt;";
</td></tr></form></table>
<h4>Správa předmětů a studentů</h4>
<table cellpadding="1">
<form method="POST" action="admin_pred.php">
<input type="hidden" name="akce" value="login">
  <tr><td align="right"><b>Heslo:</b></td>
    <td><input type="password" name="heslo"></td></tr>
  <tr><td colspan="2" align="center" height="37">
    <input type="submit" VALUE="Vstoupit &gt;&gt;";</td></tr>
</form></table></div></td></tr></table>
</div></td></tr>
<?
  } while (false);
  ODBC_Close($spojeni);
?>
  <tr><td>
    <? require "pata.php"; ?>
  </td></tr>
</table></body></html>

```

## browse.php

```

<? Header("Expires: ".GMDate("D, d M Y H:i:s")." GMT"); ?>
<html><head>
<title>Prohlížení výsledků</title>
<link rel=stylesheet type="text/css" href="style.css"></head>
<body bgcolor="#FFFFFF">
<table width="750" border="0" cellspacing="0" cellpadding="0">
  <tr><td><? require "hlava.php"; ?>
    <!-- záhlaví stránky -->
  </td></tr><tr><td><div align="center">
<!-- ----- hlavní tělo stránky ----- -->
<?
  do
  {
    @$spojeni = ODBC_Connect("cviceni", "student", "student");
    //připojení k databázi (jen pro SELECT)
    if (!$spojeni):
      echo "Spojení s databází nenavázáno :( \n";
      break;
    endif;
    //zjistění názvu aktuálního předmětu
    @$vysledek = ODBC_Exec($spojeni, "SELECT nazev
    FROM predmety
    WHERE id_pred=$id_pred");
    if (!$vysledek):
      echo "Dotaz na tabulku predmety se nezdařil :( \n";
      break;
    endif;
    if (ODBC_Fetch_Row($vysledek))
      echo "<h3>Aktuální předmět: ".ODBC_Result($vysledek, 1)."</h3>\n";
    //výpis názvu předmětu
  ?>
    <table width="" border="1" cellspacing="0" cellpadding="10"
    bgcolor="#BCC2D3">
      <tr><td height="20"><div align="center">
<!-- ----- Výběr akce ----- -->
<p><h4>Výběr akce:</h4></p>

```

```

<table border="0" cellpadding="5">
<form method="POST" action="browse.php">
  <tr>
    <td>
      <input type="radio" name="akce" value="1">
      Zobrazit všechny výsledky testu:
    <? //výpis výběru testů
      $dotaz = "SELECT *
                FROM testy
                WHERE predmet=$id_pred
                ORDER BY nazev";
      @$testy = ODBC_Exec($spojeni, $dotaz);
      if (!$testy):
        echo "Dotaz na testy se nezdařil :( \n";
        break;
      endif;
      echo "<select name=\"id_test\" size=1>\n";
      while (ODBC_Fetch_Row($testy)):
        echo "<option value=\"".ODBC_Result($testy,
'id_test')."\">".ODBC_Result($testy, "nazev")."</option>\n";
      endwhile;
      echo "</select>\n";
    ?>
  </td></tr><tr><td>
    <input type="radio" name="akce" value="2">
    Zobrazit výsledky testů studenta:
  <? //výpis výběru studentů;
    $dotaz = "SELECT prijmeni, jmeno, studenti.id_stud
              FROM studenti, pred_stud
              WHERE (studenti.id_stud=pred_stud.id_stud) AND
              id_pred=$id_pred
              ORDER BY prijmeni";
    @$studenti = ODBC_Exec($spojeni, $dotaz);
    if (!$studenti):
      echo "Dotaz na studenty se nezdařil :( \n";
      break;
    endif;
    echo "<select name=\"id_stud\" size=1>\n";
    while (ODBC_Fetch_Row($studenti)):
      echo "<option value=\"".ODBC_Result($studenti,
'id_stud')."\">".ODBC_Result($studenti, "prijmeni")."
.ODBC_Result($studenti, "jmeno")."</option>\n";
    endwhile;
    echo "</select>\n";
  ?>
  </td></tr><tr><td>
    <input type="radio" name="akce" value="3">
    Zobrazit seznam a email studentů tohoto předmětu
  </td></tr><tr><td><center>
    <input type="hidden" name="id_pred" value="<? echo $id_pred; ?>">
    <input type="submit" VALUE="Zobrazit &gt;&gt;">
  </center></td></tr></form></table>
</div></td></tr></table>
<!-- ----- Zpracování akcí ----- -->
<?
switch ($akce)
{
  case 1: // AKCE 1
    $dotaz = "SELECT nazev
              FROM testy
              WHERE id_test=$id_test";

```

```

@\$test1 = ODBC_Exec(\$spojeni, \$dotaz);
                                                    // zjištění názvu aktuálního testu
if (!$test1):
    echo "Dotaz na zjištění názvu testu se nezdařil :( \n";
    break;
endif;
if (ODBC_Fetch_Row($test1))
    echo "<h4>Výsledky: ".ODBC_Result($test1, 1)."</h4>\n";
                                                    //výpis názvu test
    \$dotaz = "SELECT prijmeni, jmeno, vysledek
              FROM studenti, vysledky
              WHERE (studenti.id_stud=vysledky.id_stud) and
(id_test=\$id_test)
              ORDER BY prijmeni";
    @\$akcel = ODBC_Exec(\$spojeni, \$dotaz);
    if (!$akcel):
        echo "Akce 1 - žádný záznam :( \n";
        break;
    endif;
    echo "<table border=\"1\" cellspacing=\"0\" cellpadding=\"4\"
bgcolor=\"#BCC2D3\">\n";
                                                    //úvod tabulky
    while (ODBC_Fetch_Row($akcel)):
        echo "<tr>\n<td>\n".ODBC_Result($akcel, "prijmeni")."
        ".ODBC_Result($akcel, "jmeno")."</td><td width=\"40\"><div
align=\"center\">".ODBC_Result($akcel,
"vysledek")."</div>\n</td>\n</tr>\n";
    endwhile;
    echo "</table>\n";
    break;
case 2:
                                                    // AKCE 2
    \$dotaz = "SELECT prijmeni, jmeno
              FROM studenti
              WHERE id_stud=\$id_stud";
    @\$stud2 = ODBC_Exec(\$spojeni, \$dotaz);
                                                    // zjištění jména aktuálního studenta
    if (!$stud2):
        echo "Dotaz na zjištění jména studenta se nezdařil :( \n";
        break;
    endif;
    if (ODBC_Fetch_Row($stud2))
        echo "<h4>Výsledky: ".ODBC_Result($stud2, 1)."
        ".ODBC_Result($stud2, 2)."</h4>\n";
                                                    //výpis názvu test
        \$dotaz = "SELECT prijmeni, jmeno, nazev, vysledek
                  FROM studenti, testy, vysledky
                  WHERE (studenti.id_stud=vysledky.id_stud) and
(vysledky.id_test=testy.id_test) and (studenti.id_stud=\$id_stud) and
(predmet=\$id_pred)";
        @\$akce2 = ODBC_Exec(\$spojeni, \$dotaz);
        if (!$akce2):
            echo "Akce 2 - žádný záznam :( \n";
            break;
        endif;
        echo "<table border=\"1\" cellspacing=\"0\" cellpadding=\"4\"
bgcolor=\"#BCC2D3\">\n";
                                                    //úvod tabulky
        while (ODBC_Fetch_Row($akce2)):
            echo "<tr>\n<td>\n".ODBC_Result($akce2, "nazev")."</td><td
width=\"40\"><div align=\"center\">".ODBC_Result($akce2,
"vysledek")."</div>\n</td>\n</tr>\n";
        endwhile;
        echo "</table>\n";
        break;

```

```

case 3: // AKCE 3
    echo "<h4>Seznam studentů v aktuálním předmětu</h4>";
    $dotaz = "SELECT *
              FROM studenti, pred_stud
              WHERE (studenti.id_stud=pred_stud.id_stud) and
(pred_stud.id_pred=$id_pred)
              ORDER BY prijmeni";
    @$akce3 = ODBC_Exec($spojeni, $dotaz);
    if (!$akce3):
        echo "Akce 3 - žádný záznam :( \n";
        break;
    endif;
    echo "<table border=\"1\" cellspacing=\"0\" cellpadding=\"4\"
bgcolor=\"#BCC2D3\">\n"; //úvod tabulky
    echo "<tr bgcolor=\"#919AB7\"><b><td><div
align=\"center\"><b>ID</b></td>\n<td><b>Příjmení a
jméno</b></td>\n<td><b>Kruh</b></td>\n<td><div align=\"center\"><b>E-
mail</b></div></td></tr>\n";
//záhlaví tabulky
    while (ODBC_Fetch_Row($akce3)):
        echo "<tr><td><div align=\"center\">\n".ODBC_Result($akce3,
"id_stud")."</div></td>\n<td>".ODBC_Result($akce3, "prijmeni")."
.ODBC_Result($akce3, "jmeno")."</td>\n<td><div
align=\"center\">".ODBC_Result($akce3, "kruh")."</div></td>\n<td><a
href=\"mailto:\".ODBC_Result($akce3, "email")."\">".ODBC_Result($akce3,
"email")."</a></td></tr>\n";
        endwhile;
        echo "</table>\n";
        break;
    }
} while (false);
ODBC_Close($spojeni);
?>
<br>
<a href="index.php">Home</a>
</div></td></tr><tr><td><? require "pata.php"; ?><!-- pata stránky -->
</td></tr></table></body></html>

```

## admin.php

```

<? Header("Expires: ".GMDDate("D, d M Y H:i:s")." GMT"); ?>
<html><head>
<title>Administrace předmětu</title>
<link rel=stylesheet type="text/css" href="style.css"></head>
<body bgcolor="#FFFFFF">
<table width="750" border="0" cellspacing="0" cellpadding="0">
<tr><td><? require "hlava.php"; ?> <!-- záhlaví stránky -->
</td></tr><tr><td><div align="center">
<?
do {
    if ($akce=="login" && $heslo=="helmut"): //ověření přístupu
        @$spojeni = ODBC_Connect("cviceni", "admin", "helmut");
        //připojení k databázi

        if (!$spojeni):
            echo "Spojení s databází nenavázáno :( \n";
            break;
        endif;

        //zjistění názvu aktuálního předmětu
        @$vysledek = ODBC_Exec($spojeni, "SELECT nazev
        FROM predmety
        WHERE id_pred=$id_pred");

        if (!$vysledek):

```

```

        echo "Dotaz na tabulku predmety se nezdařil :( \n";
        break;
    endif;
    if (ODBC_Fetch_Row($vysledek)):
        $aktpred = ODBC_Result($vysledek, 1);
        echo "<h3>Správa předmětu: $aktpred</h3>\n"; //výpis názvu předmětu
    endif;
?>
<!-- ----- PŘIDÁNÍ/ODEBRÁNÍ STUDENTŮ DO/Z PŘEDMĚTU ----- -->
    <table width="" border="1" cellspacing="0" cellpadding="10"
bgcolor="#BCC2D3">
        <tr><td height="20"><div align="center">
            <p><h4>Přidání/odebrání studenta do/z předmětu:</h4></p>
        <!-- ----- Přiradit studenta ----- -->
            <table border="0" cellpadding="5">
                <form method="POST" action="addrem_stud.php">
                    <tr><td><b>Přiradit studenta:</b></td><td>
        <?
                                                    //přiradit studenta
        $dotaz = "SELECT id_stud
                FROM pred_stud
                WHERE id_pred=$id_pred
                ORDER BY id_stud"; //dotaz na pred_stud #1
        @$vysledek = ODBC_Exec($spojeni, $dotaz);
        if (!$vysledek):
            echo "Dotaz na tab. pred_stud se nezdařil :( \n";
            break;
        endif;
        $q1 = "";
        while (ODBC_Fetch_Row($vysledek)): //naplnění proměnné pro #2
            $q1 .= ODBC_Result($vysledek, "id_stud").", ";
        endwhile;
        $q1 = SubStr($q1, 0, StrRPos($q1, ",")); //odstranění poslední čárky
        if ($q1=="")
            $q1 = 0;
        $dotaz = "SELECT id_stud, jmeno, prijmeni
                FROM studenti
                WHERE id_stud NOT IN ($q1)
                ORDER BY prijmeni"; //dotaz na stud. #2
        @$vysledek = ODBC_Exec($spojeni, $dotaz);
        if (!$vysledek):
            echo "Dotaz na tab. studenti a pred_stud se nezdařil :( \n";
            break;
        endif;
        echo "<select name=\"id_stud\" size=1>"; // SELECT
        while (ODBC_Fetch_Row($vysledek)): //výpis výběru studentů
            echo "<option value=\"\".ODBC_Result($vysledek,
'id_stud')."\">".ODBC_Result($vysledek, "prijmeni")."
".ODBC_Result($vysledek, "jmeno")."</option>\n";
        endwhile;
    ?>
        </select></td><td>
        <input type="hidden" name="from" value="add">
        <input type="hidden" name="aktpred" value="<? echo $aktpred; ?>">
        <input type="hidden" name="id_pred" value="<? echo $id_pred; ?>">
        <input type="submit" VALUE="Přiradit"></td></tr></form>
    <!-- ----- Odebrat studenta ----- -->
    <form method="POST" action="addrem_stud.php">
        <tr><td><b>Odebrat studenta:</b> </td><td>
        <select name="id_stud" size=1>
    <?
        $dotaz = "SELECT studenti.id_stud, jmeno, prijmeni

```

```

        FROM studenti, pred_stud
        WHERE (studenti.id_stud=pred_stud.id_stud) AND
id_pred=$id_pred
        ORDER BY prijmeni"; //dotaz na studenty
@$vysledek = ODBC_Exec($spojeni, $dotaz);
if (!$vysledek):
    echo "Dotaz na tab. studenti se nezdařil :( \n";
    break;
endif;
while (ODBC_Fetch_Row($vysledek)): //výpis výběru studentů
    echo "<option value=\"".ODBC_Result($vysledek,
'id_stud')."\">".ODBC_Result($vysledek, "prijmeni")."
.ODBC_Result($vysledek, "jmeno")."</option>\n";
endwhile;
?>
</select> </td><td>
<input type="hidden" name="from" value="remove">
<input type="hidden" name="aktpred" value="<? echo $aktpred; ?>">
<input type="hidden" name="id_pred" value="<? echo $id_pred; ?>">
<input type="submit" VALUE="Odebrat">
</td></tr></form></table></div></td></tr></table><br>
<!-- ----- SPRÁVA TESTŮ ----- -->
<table width="" border="1" cellspacing="0" cellpadding="10"
bgcolor="#BCC2D3"><tr><td height="20"><div align="center">
<p><h4>Testy:</h4></p>
<!-- ----- Přidat test ----- -->
<table border="0" cellpadding="5">
<form method="POST" action="insert_test.php">
<tr><td><b>Přidat test:</b> </td><td>
<input type="text" size="17" name="navez"> </td><td>
<input type="hidden" name="id_pred" value="<? echo $id_pred; ?>">
<input type="hidden" name="from" value="admin">
<input type="submit" VALUE="Přidat"></td></tr></form>
<!-- ----- Smazat test ----- -->
<form method="POST" action="delete_test.php">
<tr><td><b>Smazat test:</b> </td><td><select name="id_test" size=1>
<?
$dotaz = "SELECT *
        FROM testy
        WHERE predmet=$id_pred
        ORDER BY navez"; //dotaz na testy
@$vysledek = ODBC_Exec($spojeni, $dotaz);
if (!$vysledek):
    echo "Dotaz na testy se nezdařil :( \n";
    break;
endif;
while (ODBC_Fetch_Row($vysledek)): //výpis výběru testů
    echo "<option value=\"".ODBC_Result($vysledek,
'id_test')."\">".ODBC_Result($vysledek, "navez")."</option>\n";
endwhile;
?>
</select> </td><td>
<input type="hidden" name="from" value="admin">
<input type="hidden" name="id_pred" value="<? echo $id_pred; ?>">
<input type="submit" VALUE="Smazat">
</td></tr></form>
<!-- ----- Editovat test ----- -->
<form method="POST" action="update_test.php">
<tr><td><b>Editovat test:</b> </td><td><select name="id_test" size=1>
<?
$dotaz = "SELECT *

```





## admin\_pred.php

```
<? Header("Expires: ".GMDate("D, d M Y H:i:s")." GMT"); ?>
<html><head>
<title>Správa předmětů a studentů</title>
<link rel=stylesheet type="text/css" href="style.css"></head>
<body bgcolor="#FFFFFF">
<table width="750" border="0" cellspacing="0" cellpadding="0">
  <tr><td><? require "hlava.php"; ?>          <!-- záhlaví stránky -->
    </td></tr><tr><td><div align="center">
<?
do {
  if ($akce=="login" && $heslo=="helmut"):          //ověření přístupu
    @$spojeni = ODBC_Connect("cviceni", "admin", "helmut");
                                                    //připojení k databázi

    if (!$spojeni):
      echo "Spojení s databází nenavázáno :( \n";
      break;
    endif;
  ?>
<!-- ----- PŘEDMĚTY ----- -->
  <h3>Správa předmětů</h3>
  <table width="" border="1" cellspacing="0" cellpadding="10"
bgcolor="#BCC2D3">
  <tr><td height="20"><div align="center"><p><h4>Výběr akce:</h4></p>
<!-- ----- Přidat předmět ----- -->
  <table border="0" cellpadding="5">
  <form method="POST" action="insert_pred.php">
  <tr><td>
    <b>Přidat předmět:</b></td><td>
    <input type="text" size="17" name="nazev"> </td><td>
    <input type="hidden" name="from" value="insert_pred">
    <input type="submit" VALUE="Přidat">
  </td></tr></form>
<!-- ----- Smazat předmět ----- -->
  <form method="POST" action="delete_pred.php">
  <tr><td><b>Smazat předmět:</b> </td><td>
  <select name="id_pred" size=1>
<?
  $dotaz = "SELECT * FROM predmety ORDER BY nazev";
                                                    //dotaz na všechny předměty
  @$vysledek = ODBC_Exec($spojeni, $dotaz);
  if (!$vysledek):
    echo "Dotaz se nezdařil :( \n";
    break;
  endif;
  while (ODBC_Fetch_Row($vysledek)):          //výpis výběru předmětů
    echo "<option value=\"".ODBC_Result($vysledek,
'id_pred')."\">".ODBC_Result($vysledek, "nazev")."</option>\n";
  endwhile;
  ?>
  </select></td><td>
  <input type="hidden" name="from" value="admin_pred">
  <input type="submit" VALUE="Smazat">
  </td></tr></form>
<!-- ----- Editovat předmět ----- -->
  <form method="POST" action="update_pred.php">
  <tr><td><b>Editovat předmět:</b></td><td>
  <select name="id_pred" size=1>
<?
  $dotaz = "SELECT * FROM predmety ORDER BY nazev";
```

```

//dotaz na všechny předměty
@$vysledek = ODBC_Exec($spojeni, $dotaz);
if (!$vysledek):
    echo "Dotaz se nezdařil :( \n";
    break;
endif;
while (ODBC_Fetch_Row($vysledek)): //výpis výběru předmětů
    echo "<option value=\"".ODBC_Result($vysledek,
'id_pred')."\">".ODBC_Result($vysledek, "nazev")."</option>\n";
endwhile;
?>
</select></td><td>
<input type="hidden" name="from" value="admin_pred">
<input type="submit" VALUE="Editovat">
</td></tr></form></table></div></td></tr></table>
<!-- ----- STUDENTI ----- -->
<h3>Správa studentů</h3>
<table width="" border="1" cellspacing="0" cellpadding="10"
bgcolor="#BCC2D3">
<tr><td height="20"><div align="center"><p><h4>Výběr akce:</h4></p>
<!-- ----- Přidat studenta ----- -->
<table border="0" cellpadding="5">
<form method="POST" action="insert_stud.php">
<tr><td><b>Přidat studenta:</b>
</td><td><center><b> --> </b></center></td><td>
<input type="hidden" name="from" value="admin_stud">
<input type="submit" VALUE="Přidat">
</td></tr></form>
<!-- ----- Smazat studenta ----- -->
<form method="POST" action="delete_stud.php">
<tr><td><b>Smazat studenta:</b> </td><td>
<? //výpis výběru studentů
$dotaz = "SELECT prijmeni, jmeno, id_stud
FROM studenti
ORDER BY prijmeni";
@$studenti = ODBC_Exec($spojeni, $dotaz);
if (!$studenti):
    echo "Dotaz na studenty se nezdařil :( \n";
    break;
endif;
echo "<select name=\"id_stud\" size=1>\n";
while (ODBC_Fetch_Row($studenti)):
    echo "<option value=\"".ODBC_Result($studenti,
'id_stud')."\">".ODBC_Result($studenti, "prijmeni")."
.ODBC_Result($studenti, "jmeno")."</option>\n";
endwhile;
echo "</select>\n";
?>
</td><td>
<input type="hidden" name="from" value="admin_stud">
<input type="submit" VALUE="Smazat">
</td></tr></form>
<!-- ----- Editovat studenta ----- -->
<form method="POST" action="update_stud.php">
<tr><td><b>Editovat studenta:</b></td><td>
<? //výpis výběru studentů
$dotaz = "SELECT prijmeni, jmeno, id_stud
FROM studenti
ORDER BY prijmeni";
@$studenti = ODBC_Exec($spojeni, $dotaz);
if (!$studenti):

```

```

        echo "Dotaz na studenty se nezdařil :( \n";
        break;
    endif;
    echo "<select name=\"id_stud\" size=1>\n";
    while (ODBC_Fetch_Row($studenti)):
        echo "<option value=\"\".ODBC_Result($studenti,
'id_stud')."\">".ODBC_Result($studenti, "prijmeni")."
".ODBC_Result($studenti, "jmeno")."</option>\n";
    endwhile;
    echo "</select>\n";
?>
</td><td>
<input type="hidden" name="from" value="admin_stud">
<input type="submit" VALUE="Editovat">
</td></tr></form></table></div></td></tr></table>
<!-- ----- KONEC části LOGIN ----- -->
<?
    break;                                     //přístup nepovolen
else:
    echo "<h3>Přístup nepovolen</h3>\n";
    echo "<p><b><a href=\"index.php\">Zpět</a> na úvodní stranu.
</b></p>\n";
    break;
endif;
} while (false);
    ODBC_Close($spojeni);
?>
<!-- ----- Ověření výsledků akcí ----- to tu časem asi nebude ----- -->
<br>
<?
    echo "<h3>Šecho</h3>\n";
    $echo = "";
?>
    <a href="index.php">Home</a>
</div></td></tr><tr><td>
<? require "pata.php"; ?>                                     <!-- pata stránky -->
</td></tr></table></body></html>

```

## addrem\_stud.php

```

<? Header("Expires: ".GMDate("D, d M Y H:i:s")." GMT"); ?>
<html><head>
<title>Editace studenta</title>
<link rel=stylesheet type="text/css" href="style.css"></head>
<body bgcolor="#FFFFFF">
<table width="750" border="0" cellspacing="0" cellpadding="0">
    <tr><td><div align="center">
<?
do {
    @$spojeni = ODBC_Connect("cviceni", "admin", "helmut");
                                                //připojení k databázi

        if (!$spojeni):
            echo "Spojení s databází nenavázáno :( \n";
            break;
        endif;
    switch ($from)
    {
        case "add":
?>
        <table width="" border="1" cellspacing="0" cellpadding="10"
bgcolor="#BCC2D3">
            <tr>

```

```

        <td height="20">
            <div align="center">
<!-- ----- Přiřazení studenta do předmětu ----- -->
    <?
        $dotaz = "INSERT INTO pred_stud
                VALUES ($id_pred, $id_stud)";
        $vysledek = ODBC_Exec($spojeni, $dotaz);
        if (!$vysledek):
            echo "Studenta se nepodařilo přiřadit :( \n";
            break;
        endif;
        // přidání prázdných záznamů do tabulky výsledků
        $testy = ODBC_Exec($spojeni, "SELECT id_test FROM testy WHERE
predmet=$id_pred");
        if (!$testy):
            echo "Chyba při vkládání položek do tab. VYSLEDKY :( \n";
            break;
        endif;
        while (ODBC_Fetch_Row($testy)):
            $id_test = ODBC_Result($testy, "id_test");
            $vysled = ODBC_Exec($spojeni, "INSERT INTO vysledky VALUES ($id_test,
$id_stud, NULL)");
            if (!$vysled):
                echo "Chyba při vkládání položek do tab. VYSLEDKY (2) :( \n";
                break;
            endif;
        endwhile;
        $echo = "<h3>Student byl úspěšně přiřazen<br>do předmětu <font
color=\"magenta\">$aktpred </font></h3>";
        echo $echo;
    ?>
<form action="admin.php" > <!-- zpět form -->
<input type="hidden" name="akce" value="login">
<input type="hidden" name="heslo" value="helmut">
<input type="hidden" name="id_pred" value="<? echo $id_pred; ?>">
<input type="submit" value="Zpět na správu předmětu <? echo $aktpred; ?>">
</form>
<!-- ----- Odebrání studenta z předmětu ----- -->
    <?
        break;
        case "remove":
            //odebrání
            $dotaz = "DELETE FROM pred_stud
                WHERE id_pred=$id_pred AND id_stud=$id_stud";
            @$vysledek = ODBC_Exec($spojeni, $dotaz);
            if (!$vysledek):
                echo "Chyba, studenta se nepodařilo odebrat :( \n";
                break;
            endif;
            //vymazání příslušných řádků v tabulce VYSLEDKY
            $testy = ODBC_Exec($spojeni, "SELECT id_test FROM testy WHERE
predmet=$id_pred");
            if (!$testy):
                echo "Chyba při mazání položek z tab. VYSLEDKY :( \n";
                break;
            endif;
            while (ODBC_Fetch_Row($testy)):
                $id_test = ODBC_Result($testy, "id_test");
                $vysled = ODBC_Exec($spojeni, "DELETE FROM vysledky WHERE
id_test=$id_test AND id_stud=$id_stud");
                if (!$vysled):
                    echo "Chyba při mazání položek z tab. VYSLEDKY :( \n";
                    break;
                endif;
            endwhile;
    ?>

```

```

        endwhile;
        $echo = "<h3>Student byl úspěšně odebrán<br> z předmětu <font
color=\"magenta\">$aktpred</font></h3>";
        echo $echo;
    ?>
    <form action="admin.php">                                <!-- zpět form -->
    <input type="hidden" name="akce" value="login">
    <input type="hidden" name="heslo" value="helmut">
    <input type="hidden" name="id_pred" value="<? echo $id_pred; ?>">
    <input type="submit" value="Zpět na správu předmětu <? echo $aktpred; ?>">
    </form>
    <?
        break;
    }
    } while (false);
    ODBC_Close($spojeni);
    ?>
</div></td></tr></table></body></html>

```

### insert\_test.php

```

<? Header("Expires: ".GMMDate("D, d M Y H:i:s")." GMT"); ?>
    <html><head><title>Smazání testu</title></head><body>
    <?                                                    //Přidat test
do {
    if ($from=="admin"):                                //ověření přístupu
        $echo = "Test<font color=\"magenta\"> $nazev </font>byl úspěšně přidán.";
        @$spojeni = ODBC_Connect("cviceni", "admin", "helmut");
                                                    //připojení k databázi

        if (!$spojeni):
            $echo = "Chyba, test se nepodařilo přidat.";
            break;
        endif;
        $dotaz = "SELECT Max(id_test)+1 FROM testy";    //generování nového ID
        @$vysledek = ODBC_Exec($spojeni, $dotaz);
        if (!$vysledek):
            $echo = "Chyba, test se nepodařilo přidat.";
            break;
        endif;
        if (ODBC_Fetch_Row($vysledek))                //přečtení nového ID
            $id_test = ODBC_Result($vysledek, 1);
        else {
            $echo = "Chyba, test se nepodařilo přidat.";
            break;
        }
        if ($id_test=="")
            $id_test = 1;
        $dotaz = "INSERT INTO testy
                VALUES ($id_test, '$nazev', $id_pred)";    //vložení testu
        @$vysledek = ODBC_Exec($spojeni, $dotaz);
        if (!$vysledek):
            $echo = "Chyba, test se nepodařilo přidat.";
            break;
        endif; // přidání záznamů do tabulky výsledků pro příslušné studenty
        $stud = ODBC_Exec($spojeni, "SELECT id_stud
                FROM pred_stud
                WHERE id_pred=$id_pred");    //?ID
    if (!$stud):
        echo "Chyba při vkládání položek do tab. VYSLEDKY :( \n";
        break;
    endif;
    while (ODBC_Fetch_Row($stud)):

```

```

        $id_stud = ODBC_Result($stud, "id_stud");
        $vysled = ODBC_Exec($spojeni, "INSERT INTO vysledky VALUES ($id_test,
$id_stud, NULL)");
        if (!$vysled):
            echo "Chyba při vkládání položek do tab. VYSLEDKY (2) :( \n";
            break;
        endif;
    endwhile;
    break; //přístup nepovoleno
else:
    $echo = "Chyba, test se nepodařilo přidat.";
    break;
endif;
} while (false);
    ODBC_Close($spojeni);
?>
<!-- ----- Informace o provedení ----- -->
<table width="750" border="0" cellspacing="0" cellpadding="0">
<tr><td><div align="center">
<table width="" border="1" cellspacing="0" cellpadding="10"
bgcolor="#BCC2D3">
    <tr><td height="20"><div align="center">
        <h3><? echo $echo; ?></h3>
        <form action="admin.php">
            <input type="hidden" name="akce" value="login">
            <input type="hidden" name="heslo" value="helmut">
            <input type="hidden" name="id_pred" value="<? echo $id_pred; ?>">
            <input type="submit" value="Zpět na správu předmětu">
        </div></td></form></tr></table></div></td></tr></table></body></html>

```

## delete\_test.php

```

<? Header("Expires: ".GMDDate("D, d M Y H:i:s")." GMT"); ?>
<html><head><title>Smazání testu</title></head><body>
<? //Smazat předmět
do {
    @$spojeni = ODBC_Connect("cviceni", "admin", "helmut"); //připojení
    if (!$spojeni):
        echo "Chyba, spojení a databází.\n";
        break;
    endif;
    $dotaz = "SELECT nazev
        FROM testy
        WHERE id_test=$id_test"; //přečtení názvu testu
    @$vysledek = ODBC_Exec($spojeni, $dotaz);
    if (!$vysledek):
        echo "Chyba - nenalezeno jméno testu.\n";
        break;
    endif;
    if (ODBC_Fetch_Row($vysledek))
        $nazev = ODBC_Result($vysledek, 1);
    else {
        echo "Chyba - nenalezeno jméno testu.\n";
        break;
    } ?>
<table width="750" border="0" cellspacing="0" cellpadding="0">
<tr><td><div align="center">
<?
switch ($from)
{
    case "admin": //ověření smazání
?>

```

```

<!-- ----- Ověření smazání testu ----- -->





```

## update\_test.php

```

<? Header("Expires: ".GMDate("D, d M Y H:i:s")." GMT"); ?>

```



```

<html><head><title>Editace testu</title>
<link rel=stylesheet type="text/css" href="style.css">
</head><body bgcolor="#FFFFFF">
<table width="750" border="0" cellspacing="0" cellpadding="0">
<tr><td><div align="center">
<!-- ----- hlavní tělo stránky ----- -->
<?
do {
    @$spojeni = ODBC_Connect("cviceni", "admin", "helmut"); //připojení
    if (!$spojeni):
        echo "Spojení s databází nenavázáno :( \n";
        break;
    endif;
    switch ($from)
    {
        case "admin":
            ?>
                <table width="" border="1" cellspacing="0" cellpadding="10"
bgcolor="#BCC2D3">
                    <tr><td height="20"><div align="center">
<!-- ----- Zobrazení akce ----- -->
<?
                @$vysledek = ODBC_Exec($spojeni, "SELECT *
                                                FROM testy
                                                WHERE id_test=$id_test");
                if (!$vysledek):
                    echo "Dotaz na tabulku testy se nezdařil :( \n";
                    break;
                endif;
                if (ODBC_Fetch_Row($vysledek))
                    $nazev = ODBC_Result($vysledek, "nazev"); //výpis názvu předmětu
                echo "<h3>Editace testu: $nazev </h3>\n";
            ?>
            <form method="POST" action="update_test.php">
            <table width="0" border="0" cellspacing="0" cellpadding="5">
                <tr><td><b>ID:</b></td>
                <td><? echo $id_test; ?></td>
            </tr><tr><td><b>Název:</b></td><td>
            <input type="text" name="nazev2" value="<? echo $nazev; ?>">
            <input type="hidden" name="nazev" value="<? echo $nazev; ?>">
            <input type="hidden" name="id_pred" value="<? echo $id_pred; ?>">
            <input type="hidden" name="id_test" value="<? echo $id_test; ?>">
            <input type="hidden" name="from" value="update_test">
            </td></tr><tr><td colspan="2"><center>
            <input type="submit" VALUE="Změnit"></center></td></tr></form></table>
            </div></td></tr></table>
            <?
                break; //změna - UPDATE
            case "update_test":
                ?>
<!-- ----- Zpracování akce ----- -->
                <table width="" border="1" cellspacing="0" cellpadding="10"
bgcolor="#BCC2D3"><tr><td height="20"><div align="center">
                <?
                    $dotaz = "UPDATE testy
                            SET nazev = '$nazev2'
                            WHERE id_test=$id_test";
                    @$vysledek = ODBC_Exec($spojeni, $dotaz);
                    if (!$vysledek):
                        echo "Změna neprovedena :( \n";
                        break;

```

```

        else:
            $echo = "<h3>Změna názvu testu z <font
color=\\"magenta\\">$nazev</font><br>na<font color=\\"magenta\\"> $nazev2
</font>provedena.</h3>";
            endif;
            echo $echo;
        ?>

        <form action="admin.php">
        <input type="hidden" name="akce" value="login">
        <input type="hidden" name="heslo" value="helmut">
        <input type="hidden" name="id_pred" value="<? echo $id_pred; ?>">
        <input type="submit" value="Zpět na správu testů">
        </div></td></tr></table></form>

    <?
        break;
    }
} while (false);
ODBC_Close($spojeni);
?>
</div></td></tr></table></body></html>

```

## vysled.php

```

<? Header("Expires: ".GMDdate("D, d M Y H:i:s")." GMT"); ?>
<html><head><title>Editace výsledků</title>
<link rel=stylesheet type="text/css" href="style.css"></head>
<body bgcolor="#FFFFFF">
<table width="750" border="0" cellspacing="0" cellpadding="0">
<tr><td><div align="center">
<?
do {
    @$spojeni = ODBC_Connect("cviceni", "admin", "helmut"); //připojení
    if (!$spojeni):
        echo "Spojení s databází nenavázáno :( \n";
        break;
    endif;
    switch ($from)
    {
        case "admin":
        ?>
        <table width="" border="1" cellspacing="0" cellpadding="10"
bgcolor="#BCC2D3">
        <tr><td height="20"><div align="center">
<!-- ----- Editace ----- -->
        <?
        //načtení výsledků
        @$nazev = ODBC_Exec($spojeni, "SELECT nazev FROM testy WHERE
id_test=$id_test");
        if (!$nazev):
            echo "Název aktuálního testu nezjištěn.";
            break;
        endif;
        if (ODBC_Fetch_Row($nazev))
            $akt_nazev = ODBC_Result($nazev, "nazev");// výpis akt. názvu testu
        $dotaz = "SELECT studenti.id_stud, jmeno, prijmeni, vysledek
        FROM studenti, vysledky
        WHERE studenti.id_stud=vysledky.id_stud AND id_test=$id_test
        ORDER BY prijmeni, jmeno, kruh";
        @$result = ODBC_Exec($spojeni, $dotaz);
        if (!$result):
            echo "Dotaz na tabulku vysledky (+ studenti) se nezdařil.";
            break;
        endif;

```

```

?>
    <form method="POST" action="vysled.php">      <!-- vstupní formulář -->
    <table width="" border="0" cellspacing="0" cellpadding="5">
    <h4>Editace výsledků: <? echo $aktnazev; ?></h4>
    <?
    echo "<table width=\"\" border=\"0\" cellspacing=\"0\"
cellpadding=\"5\">"; //výpis jmen a text. polí form.
    while (ODBC_Fetch_Row($result)): //tabulka
        echo "<tr><td><b>".ODBC_Result($result, "prijmeni")."
".ODBC_Result($result, "jmeno")."</b></td><td><input type=\"text\"
size=\"10\" name=\"".ODBC_Result($result, "id_stud").\"
value=\"".ODBC_Result($result, "vysledek").\"></td></tr>";
    endwhile;
?>
    <tr><td colspan="2"><center>
    <input type="hidden" name="from" value="vysled">
    <input type="hidden" name="id_pred" value="<? echo $id_pred; ?>">
    <input type="hidden" name="aktnazev" value="<? echo $aktnazev; ?>">
    <input type="hidden" name="id_test" value="<? echo $id_test; ?>">
    <input type="submit" VALUE="Potvrdit">
    </center></td></tr></table></form>
    <form action="admin.php"> <!--Zpět form -->
    <input type="hidden" name="akce" value="login">
    <input type="hidden" name="heslo" value="helmut">
    <input type="hidden" name="id_pred" value="<? echo $id_pred; ?>">
    <input type="submit" value="Zpět"></table></form>
<!-- ----- Zpracování akce ----- -->
<?
    break;
    case "vysled": //EDITACE
        $echo = "<h3>Změna výsledků testu <font
color=\"magenta\">$aktnazev</font> provedena.</h3>";
        $dotaz = "SELECT studenti.id_stud
FROM studenti, pred_stud
WHERE studenti.id_stud=pred_stud.id_stud AND
id_pred=$id_pred
ORDER BY prijmeni, jmeno, kruh";
        @$stud = ODBC_Exec($spojeni, $dotaz);
        if (!$stud):
            $echo = "Chyba při změně výsledků :( \n";
            break;
        endif;
        while (ODBC_Fetch_Row($stud)):
            $id_stud = ODBC_Result($stud, "id_stud");
            ${$id_stud} += 0; //převod na číslo
            $vysled = ODBC_Exec($spojeni, "UPDATE vysledky
SET vysledek=${$id_stud}
WHERE id_test=$id_test AND
id_stud=$id_stud");
            if (!$vysled):
                $echo = "Chyba při změně výsledků (2) :( \n";
                break;
            endif;
        endwhile;
        echo $echo;
?>
    <form action="admin.php"> <!-- zpět form -->
    <input type="hidden" name="akce" value="login">
    <input type="hidden" name="heslo" value="helmut">
    <input type="hidden" name="id_pred" value="<? echo $id_pred; ?>">
    <input type="submit" value="Zpět na správu předmětu"></form>

```

```

<?
    break;
}
} while (false);
ODBC_Close($spojeni);
?>
</div></td></tr></table></body></html>

```

## insert\_pred.php

```

<? Header("Expires: ".GMDate("D, d M Y H:i:s")." GMT"); ?>
<?
    //Přidat předmět
do
{
    if ($from=="insert_pred"):
        //ověření přístupu
        $echo = "Předmět <font color=\\"magenta\\"> $nazev </font> byl úspěšně
přidán.";
        @$spojeni = ODBC_Connect("cviceni", "admin", "helmut");
        //připojení k databázi

        if (!$spojeni):
            $echo = "Chyba, předmět se nepodařilo přidat.";
            break;
        endif;
        $dotaz = "SELECT Max(id_pred)+1 FROM predmety";
        //nové ID
        @$vysledek = ODBC_Exec($spojeni, $dotaz);
        if (!$vysledek):
            $echo = "Chyba, nové ID nezjištěno.";
            break;
        endif;
        if (ODBC_Fetch_Row($vysledek))
            //přečtení ID
            $id_pred = ODBC_Result($vysledek, 1);
        else {
            $echo = "Chyba, nové ID nezjištěno.";
            break;
        }
        if ($id_pred=="")
            $id_pred = 1;
        $dotaz = "INSERT INTO predmety
            VALUES ($id_pred, '$nazev')";
        //vložení předmětu
        @$vysledek = ODBC_Exec($spojeni, $dotaz);
        if (!$vysledek):
            $echo = "Chyba, předmět se nepodařilo přidat.";
            break;
        endif;
        break;
        //přístup nepovolen
    else:
        $echo = "Chyba, předmět se nepodařilo přidat.";
        break;
    endif;
} while (false);
ODBC_Close($spojeni);
?>
<html><head><title>Přidání předmětu</title>
<link rel=stylesheet type="text/css" href="style.css"></head>
<body bgcolor="#FFFFFF">
<table width="750" border="0" cellspacing="0" cellpadding="0">
<tr><td><div align="center">
<table width="" border="1" cellspacing="0" cellpadding="10"
bgcolor="#BCC2D3">
<tr align="center"><td><h4><? echo $echo; ?> </h4>
<form action="admin_pred.php">
    <!-- zpět form -->
    <input type="hidden" name="akce" value="login">

```

```



</form></td></tr></table></div></td></tr></table></body></html>

```

## delete\_pred.php

```

<? Header("Expires: ".GMMDate("D, d M Y H:i:s")." GMT"); ?>
<html><head><title>Smazání předmětu</title></head><body>
<table width="750" border="0" cellspacing="0" cellpadding="0">
<tr><td><div align="center">
<table width="" border="1" cellspacing="0" cellpadding="10"
bgcolor="#BCC2D3"><tr align="center"><td>
<? //Smazat předmět
do {
@$spojeni = ODBC_Connect("cviceni", "admin", "helmut"); //připojení
if (!$spojeni):
echo "Chyba, spojení a databáží.\n";
break;
endif;
$dotaz = "SELECT nazev
FROM predmety
WHERE id_pred=$id_pred"; //přečtení názvu předmětu
@$vysledek = ODBC_Exec($spojeni, $dotaz);
if (!$vysledek):
echo "Chyba - nenalezeno jméno předmětu\n";
break;
endif;
if (ODBC_Fetch_Row($vysledek))
$nazev = ODBC_Result($vysledek, 1);
else {
echo "Chyba - nenalezeno jméno předmětu\n";
break;
}
}
switch ($from)
{
case "admin_pred": //ověření smazání
?>
<!-- ----- Ověření smazání předmětu ----- -->
<h4>Opravdu si přejete SMAZAT předmět <font color="magenta"> <? echo
$nazev; ?> </font><br>včetně všech součástí v tabulce testů, výsledků a
přiřazení studentů ?</h4>
<table cellpadding="5"><tr><td>
<form method="POST" action="delete_pred.php">
<input type="hidden" name="id_pred" value="<? echo $id_pred; ?>">
<input type="hidden" name="from" value="delete_pred">
<input type="hidden" name="nazev" value="<? echo $nazev; ?>">
<input type="submit" value="Ano"></form></td>
<td><form action="admin_pred.php">
<input type="hidden" name="akce" value="login">
<input type="hidden" name="heslo" value="helmut">
<input type="submit" value="Zpět"></form>
<?
break;
case "delete_pred": //smazání
?>
<!-- ----- Smazání předmětu ----- -->
<?
$dotaz = "DELETE FROM predmety
WHERE id_pred=$id_pred"; //mazání z tab PREDMETY
@$vysledek = ODBC_Exec($spojeni, $dotaz);
if (!$vysledek):
echo "Chyba při mazání předmětu (tab. predmety).\n";

```

```

        break;
    endif;
    $dotaz = "DELETE FROM pred_stud
              WHERE id_pred=$id_pred";           //mazání z tab PRED_STUD
    @$vysledek = ODBC_Exec($spojeni, $dotaz);
    if (!$vysledek):
        echo "Chyba při mazání předmětu (tab. pred_stud).\n";
        break;
    endif;
    $dotaz = "SELECT id_test
              FROM testy
              WHERE predmet=$id_pred";           //mazání z výsledků
    @$testy = ODBC_Exec($spojeni, $dotaz);
    if (!$testy):
        echo "Dotaz na testy se nezdařil :( \n";
        break;
    endif;
    while (ODBC_Fetch_Row($testy)):
        $id_test = ODBC_Result($testy, "id_test");
        @$vysledek = ODBC_Exec($spojeni, "DELETE FROM vysledky WHERE
id_test=$id_test");
        if (!$vysledek):
            echo "Chyba při mazání předmětu (tab. vysledky - test:
$id_test).\n";
            break;
        endif;
    endwhile;
    $dotaz = "DELETE FROM testy
              WHERE predmet=$id_pred";           //mazání z tab TESTY
    @$vysledek = ODBC_Exec($spojeni, $dotaz);
    if (!$vysledek):
        echo "Chyba při mazání předmětu (tab. testy).\n";
        break;
    endif;
    $echo = "<h3>Předmět<font color=\"magenta\"> $navez </font> byl úspěšně
smazán.</h3>\n";
    echo $echo;
    ?>
        <form action="admin_pred.php">
        <input type="hidden" name="akce" value="login">
        <input type="hidden" name="heslo" value="helmut">
        <input type="submit" value="Zpět na úpravu předmětů">
    <?
        break;
    }
    } while (false);
    ODBC_Close($spojeni);
    ?>
    </td></form></tr></table></div></td></tr></table></body></html>

```

## update\_pred.php

```

<? Header("Expires: ".GMDate("D, d M Y H:i:s")." GMT"); ?>
<html><head><title>Editace předmětu</title>
<link rel=stylesheet type="text/css" href="style.css"></head>
<body bgcolor="#FFFFFF">
<table width="750" border="0" cellspacing="0" cellpadding="0">
<tr><td><div align="center">
<!-- ----- hlavní tělo stránky ----- -->
<?
    do {
        @$spojeni = ODBC_Connect("cviceni", "admin", "helmut"); //připojení

```

```

        if (!$spojeni):
            echo "Spojení s databází nenavázáno :( \n";
            break;
        endif;
    switch ($from)
    {
        case "admin_pred":
            ?>
                <table width="" border="1" cellspacing="0" cellpadding="10"
bgcolor="#BCC2D3">
                    <tr><td height="20"><div align="center">
<!-- ----- Výběr akce ----- -->
                <?
                    @$vysledek = ODBC_Exec($spojeni, "SELECT *
                                                    FROM predmety
                                                    WHERE id_pred=$id_pred");

                    if (!$vysledek):
                        echo "Dotaz na tabulku predmety se nezdařil :( \n";
                        break;
                    endif;
                    if (ODBC_Fetch_Row($vysledek))
                        $nazev = ODBC_Result($vysledek, "nazev"); //výpis názvu
                        echo "<h3>Editovat předmět: $nazev </h3>\n";
                    ?>
                <form method="POST" action="update_pred.php">
                <table width="0" border="0" cellspacing="0" cellpadding="5">
                    <tr><td><b>ID:</b></td>
                        <td><? echo $id_pred; ?></td></tr><tr>
                        <td><b>Název:</b></td><td>
                            <input type="text" name="nazev2" value="<? echo $nazev; ?>">
                            <input type="hidden" name="nazev" value="<? echo $nazev; ?>">
                            <input type="hidden" name="id_pred" value="<? echo $id_pred; ?>">
                            <input type="hidden" name="from" value="update_pred"></td></tr>
                    <tr><td colspan="2"><center>
                        <input type="submit" VALUE="Změnit">
                    </center></td></tr></table></form>
                    </div></td></tr></table>
<!-- ----- Zpracování akcí ----- -->
                <p>&nbsp;</p>
                <?
                    break; //změna - UPDATE
                case "update_pred":
                    $dotaz = "UPDATE predmety
                            SET nazev = '$nazev2'
                            WHERE id_pred=$id_pred";
                    @$vysledek = ODBC_Exec($spojeni, $dotaz);
                    if (!$vysledek):
                        echo "Změna neprovedena :( \n";
                        break;
                    else:
                        $echo = "<h3>Změna názvu předmětu z <font
color=\"magenta\">$nazev</font><br>na<font color=\"magenta\"> $nazev2
</font>provedena.</h3>";
                    endif;
                    echo $echo;
                ?>
                <form action="admin_pred.php">
                <input type="hidden" name="akce" value="login">
                <input type="hidden" name="heslo" value="helmut">
                <input type="submit" value="Zpět na úpravu předmětů"></form>
                <?

```

```

        break;
    }
} while (false);
ODBC_Close($spojeni);
?>
</div></td></tr></table></body></html>

```

## insert\_stud.php

```

<? Header("Expires: ".GMDate("D, d M Y H:i:s")." GMT"); ?>
<html><head><title>Přidání studenta</title>
<link rel=stylesheet type="text/css" href="style.css"></head>
<body bgcolor="#FFFFFF">
<table width="750" border="0" cellspacing="0" cellpadding="0">
<tr><td><div align="center">
<!-- ----- hlavní tělo stránky ----- -->
<?
do {
    @$spojeni = ODBC_Connect("cviceni", "admin", "helmut"); //připojení
    if (!$spojeni):
        echo "Spojení s databází nenavázáno :( \n";
        break;
    endif;
    switch ($from)
    {
    case "admin_stud":
?>
        <table width="" border="1" cellspacing="0" cellpadding="10"
bgcolor="#BCC2D3">
        <tr><td height="20"><div align="center">
        <h4>Přidání nového studenta</h4>
<!-- ----- Akce ----- -->
<?
    $dotaz = "SELECT Max(id_stud)+1 FROM studenti"; // nové ID
    @$vysledek = ODBC_Exec($spojeni, $dotaz);
    if (!$vysledek):
        echo "Chyba, nové ID nezjištěno.";
        break;
    endif;
    if (ODBC_Fetch_Row($vysledek)) //přečtení nového ID
        $id_stud = ODBC_Result($vysledek, 1);
    else {
        echo "Chyba, nové ID nezjištěno.";
        break;
    }
    if ($id_stud=="") //je-li první ID=1
        $id_stud = 1;
?>
    <form method="POST" action="insert_stud.php"> <!-- vstupní formulář -->
    <table width="0" border="0" cellspacing="0" cellpadding="5">
    <tr><td><b>ID:</b></td>
    <td><b><? echo $id_stud; ?></b>
    <input type="hidden" name="id_stud" value="<? echo $id_stud; ?>">
    </td></tr>
    <tr><td><b>Jméno:</b></td>
    <td><input type="text" name="jmeno" size="25"></td></tr>
    <tr><td><b>Příjmení:</b></td>
    <td><input type="text" name="prijmeni" size="25"></td></tr>
    <tr><td><b>Kruh:</b></td>
    <td><input type="text" name="kruh" size="10"></td></tr>
    <tr><td><b>E-mail:</b></td>
    <td><input type="text" name="email" size="30"></td></tr>

```



```

<tr><td colspan="2"><center>
  <input type="hidden" name="from" value="insert_stud">
  <input type="submit" VALUE="Přidat"></center></td></tr></form></table>
</div></td></tr></table>
<!-- ----- Zpracování akce ----- -->
<?
  break;
case "insert_stud":                                     //přidání - INSERT
  $dotaz = "INSERT INTO studenti
            VALUES ($id_stud, '$jmeno', '$prijmeni', $kruh, '$email')";
  @$vysledek = ODBC_Exec($spojeni, $dotaz);
  if (!$vysledek):
    echo "Chyba, studenta se nepodařilo přidat.";
    break;
  else:
    $echo = "<h3>Student <font color=\"magenta\">$jmeno $prijmeni
</font>přidán.</h3>";
  endif;
  echo $echo;
?>
  <form action="admin_pred.php">                                     <!-- zpět form -->
  <input type="hidden" name="akce" value="login">
  <input type="hidden" name="heslo" value="helmut">
  <input type="submit" value="Zpět na správu studentů"></form>
  <form action="insert_stud.php">                                     <!-- přidat další -->
  <input type="hidden" name="from" value="admin_stud">
  <input type="submit" value="Přidat dalšího studenta">
  </form>
<?
  break;
}
} while (false);
ODBC_Close($spojeni);
?>
</div></td></tr></table></body></html>

```

## delete\_stud.php

```

<? Header("Expires: ".GMMDate("D, d M Y H:i:s")." GMT"); ?>
<html><head><title>Přidání studenta</title>
<link rel=stylesheet type="text/css" href="style.css"></head>
<body bgcolor="#FFFFFF">
<table width="750" border="0" cellspacing="0" cellpadding="0">
<tr><td><div align="center">
<!-- ----- hlavní tělo stránky ----- -->
<?
  do {
    @$spojeni = ODBC_Connect("cviceni", "admin", "helmut"); //připojení
    if (!$spojeni):
      echo "Spojení s databází nenavázáno :( \n";
      break;
    endif;
  switch ($from)
  {
    case "admin_stud":
  ?>
    <table width="" border="1" cellspacing="0" cellpadding="10"
bgcolor="#BCC2D3">
      <tr><td height="20"><div align="center">
        <h4>Opravdu chcete SMAZAT studenta<br>včetně všech součástí?</h4>
  <!-- ----- Potvrzení akce ----- -->
  <?

```

```

$dotaz = "SELECT *
          FROM studenti
          WHERE id_stud=$id_stud";           //ověření smazání
@$vysledek = ODBC_Exec($spojeni, $dotaz);
if (!$vysledek):
    echo "Dotaz na tabulku studenti se nezdařil.";
    break;
endif;
if (ODBC_Fetch_Row($vysledek)):
    $id_stud = ODBC_Result($vysledek, "id_stud");
    $jmeno = ODBC_Result($vysledek, "jmeno");
    $prijmeni = ODBC_Result($vysledek, "prijmeni");
    $kruh = ODBC_Result($vysledek, "kruh");
    $email = ODBC_Result($vysledek, "email");
else:
    echo "Dotaz na tabulku studenti se nezdařil.";
    break;
endif;
?>
<form method="POST" action="delete_stud.php"> <!-- vstupní formulář -->
<table width="0" border="0" cellspacing="0" cellpadding="5">
    <tr><td><b>ID:</b></td>
        <td><b>?<? echo $id_stud; ?></b></td></tr>
    <tr><td><b>Jméno:</b></td>
        <td><b>?<? echo $jmeno; ?></b></td></tr>
    <tr><td><b>Příjmení:</b></td>
        <td><b>?<? echo $prijmeni; ?></b></td></tr>
    <tr><td><b>Kruh:</b></td>
        <td><b>?<? echo $kruh; ?></b></td></tr>
    <tr><td><b>E-mail:</b></td>
        <td><b>?<? echo $email; ?></b></td></tr>
    <tr><td colspan="2"><center>
        <input type="hidden" name="from" value="delete_stud">
        <input type="hidden" name="id_stud" value="?<? echo $id_stud; ?>">
        <input type="hidden" name="jmeno" value="?<? echo $jmeno; ?>">
        <input type="hidden" name="prijmeni" value="?<? echo $prijmeni; ?>">
        <input type="submit" VALUE="Smazat"> </center></td></tr></form>
    <tr><td colspan="2"> <center>
        <form action="admin_pred.php">           <!-- zpět form -->
        <input type="hidden" name="akce" value="login">
        <input type="hidden" name="heslo" value="helmut">
        <input type="submit" value="Zpět"></center></td></form></tr></table>
</div></td></tr></table>
<!-- ----- Zpracování akce ----- -->
<?
    break;
case "delete_stud":
    $dotaz = "DELETE FROM studenti           //mazání - DELETE
            WHERE id_stud=$id_stud";       //mazání z tab STUDENTI
@$vysledek = ODBC_Exec($spojeni, $dotaz);
if (!$vysledek):
    echo "Chyba při mazání studenta (tab. studenti).\n";
    break;
endif;
$dotaz = "DELETE FROM pred_stud
          WHERE id_stud=$id_stud";         //mazání z tab PRED_STUD
@$vysledek = ODBC_Exec($spojeni, $dotaz);
if (!$vysledek):
    echo "Chyba při mazání studenta (tab. pred_stud).\n";
    break;
endif;

```

```

$dotaz = "DELETE FROM vysledky
          WHERE id_stud=$id_stud";          //mazání z tab VYSLEDKY
@$vysledek = ODBC_Exec($spojeni, $dotaz);
if (!$vysledek):
    echo "Chyba při mazání studenta (tab. pred_stud).\n";
    break;
endif;
$echo = "<h3>Student<font color=\"magenta\"> $prijmeni $jmeno </font>
byl úspěšně smazán.</h3>\n";
echo $echo;
?>
    <form action="admin_pred.php"                                <!-- zpět form -->
    <input type="hidden" name="akce" value="login">
    <input type="hidden" name="heslo" value="helmut">
    <input type="submit" value="Zpět na správu studentů">
    </form>
<?
    break;
}
} while (false);
ODBC_Close($spojeni);
?>
</div></td></tr></table></body></html>

```

## update\_stud.php

```

<? Header("Expires: ".GMDate("D, d M Y H:i:s")." GMT"); ?>
<html><head><title>Editace studenta</title>
<link rel=stylesheet type="text/css" href="style.css"></head>
<body bgcolor="#FFFFFF">
<table width="750" border="0" cellspacing="0" cellpadding="0">
<tr><td><div align="center">
<!-- ----- hlavní tělo stránky ----- -->
<?
    do {
        @$spojeni = ODBC_Connect("cviceni", "admin", "helmut"); //připojení
        if (!$spojeni):
            echo "Spojení s databází nenavázáno :( \n";
            break;
        endif;
    switch ($from)
    {
        case "admin_stud":
            ?>
                <table width="" border="1" cellspacing="0" cellpadding="10"
                bgcolor="#BCC2D3">
                    <tr><td height="20"><div align="center">
<!-- ----- Potvrzení editace ----- -->
<?
                    $dotaz = "SELECT *
                              FROM studenti
                              WHERE id_stud=$id_stud";          //ověření smazání
@$vysledek = ODBC_Exec($spojeni, $dotaz);
if (!$vysledek):
    echo "Dotaz na tabulku studenti se nezdařil.";
    break;
endif;
if (ODBC_Fetch_Row($vysledek)):
    $id_stud = ODBC_Result($vysledek, "id_stud");
    $jmeno = ODBC_Result($vysledek, "jmeno");
    $prijmeni = ODBC_Result($vysledek, "prijmeni");
    $kruh = ODBC_Result($vysledek, "kruh");

```

```

    $email = ODBC_Result($vysledek, "email");
else:
    echo "Dotaz na tabulku studenti se nezdařil.";
    break;
endif;
?>
<h4>Editace studenta</h4>
<form method="POST" action="update_stud.php"> <!-- vstupní formulář -->
<table width="0" border="0" cellspacing="0" cellpadding="5">
  <tr><td><b>ID:</b></td>
    <td><b><? echo $id_stud; ?></b>
      <input type="hidden" name="id_stud" value="<? echo $id_stud; ?>"
    </td></tr>
  <tr><td><b>Jméno:</b></td>
    <td><input type="text" name="jmeno" size="25" value="<? echo
$jmeno; ?>"></td></tr>
  <tr><td><b>Příjmení:</b></td>
    <td><input type="text" name="prijmeni" size="25" value="<? echo
$prijmeni; ?>"></td></tr>
  <tr><td><b>Kruh:</b></td>
    <td><input type="text" name="kruh" size="10" value="<? echo $kruh;
?>"></td></tr>
  <tr><td><b>E-mail:</b></td>
    <td><input type="text" name="email" size="30" value="<? echo
$email; ?>"></td></tr>
  <tr><td colspan="2"><center>
    <input type="hidden" name="from" value="update_stud">
    <input type="submit" VALUE="Změnit"></center></td></tr></form></table>
</div></td></tr></table>
<!-- ----- Zpracování akce ----- -->
<?
  break;
case "update_stud": //změna - UPDATE
  $dotaz = "UPDATE studenti
            SET jmeno='$jmeno', prijmeni='$prijmeni', kruh=$kruh,
email='$email'
            WHERE id_stud=$id_stud"; //změna údajů studenta
  @$vysledek = ODBC_Exec($spojeni, $dotaz);
  if (!$vysledek):
    echo "Chyba, studenta se nepodařilo změnit.";
    break;
  else:
    $echo = "<h3>Změna údajů studenta (<font color=\"magenta\">$jmeno
$prijmeni </font>) provedena.</h3>";
  endif;
  echo $echo;
?>
  <form action="admin_pred.php"> <!-- zpět form -->
  <input type="hidden" name="akce" value="login">
  <input type="hidden" name="heslo" value="helmut">
  <input type="submit" value="Zpět na správu studentů"></form>
<?
  break;
}
} while (false);
ODBC_Close($spojeni);
?>
</div></td></tr></table></body></html>

```