# SYSTEM FOR THE AUTOMATED CONTROL OF DATABASE TASK

*Tomáš VÁŇA - David ŽÁK - Jiří ZECHMEISTER - Jiří LEBDUŠKA*

***Abstract:*** This article deals with improving the quality and effectiveness of database lessons. It focuses primarily on the automation of student's work control. The paper describes an existing functional application that is used for teaching database courses at the Faculty of Electrical Engineering and Informatics at the University of Pardubice. The application is mainly designed for teachers. Teachers use the application for students' work verifying, but students can also use this application to verify their own work. This article describes possibilities of using this application during database lessons, for checking home works and for credit and exam checking

***Key words:*** database, Oracle, teaching of databases courses, automatic control, education, practical lessons, *web application.*

## SYSTÉM PRO AUTOMATIZOVANOU KONTROLU DATABÁZOVÝCH ÚLOH

***Resumé:*** Článek se zabývá problematikou zvýšení kvality a efektivity výuky databázových předmětů. Zaměřuje se především na automatizaci kontroly praktických úloh. Článek popisuje již konkrétní fungující aplikaci, která je využívána v rámci výuky databázových předmětů na Fakultě elektrotechniky a informatiky Univerzity Pardubice. Daná aplikace slouží nejen pedagogům, kterým pomáhá získat přehled o řešení jednotlivých úloh studenty, ale zároveň slouží i samotným studentům k okamžité kontrole jejich práce. Jsou popsány možnosti využití aplikace pro automatizovanou kontrolu práce studentů ve výuce při cvičeních, kontrolu domácí přípravy a praktických úloh řešených během zápočtů a zkoušek.

***Klíčová slova:*** databáze, Oracle, výuka databázových předmětů, automatizace kontroly, vzdělávání, praktická výuka, webová aplikace.

## 1 Úvod

Nebývalý rozvoj informačních technologií v posledních letech vedl k nutnosti uchovávat stále větší objemy dat efektivnějšími způsoby. Na trhu se objevila databázová řešení mnoha různých společností. Tato řešení si nalezla cestu do velkého počtu aplikací a systémů. Z databází se staly komplexní a sofistikované nástroje pro ukládání a manipulaci s daty. Pro využití takto komplexních systémů je zapotřebí odborníků se širokou škálou znalostí a dovedností. Zvýšila se tak poptávka po odbornících z řad databázových specialistů, ať už se jedná o specialisty na konkrétní databázový systém či specialisty s výbornou znalostí jazyka SQL [1], který je standardním dotazovacím jazykem podporovaným prakticky všemi známými relačními databázovými systémy.

Výše uvedené trendy stály za zásadní proměnou výuky [2] databázových systému jak na vysokých školách, tak i na odborných středních školách. Databázově orientované předměty si již vydobyly své pevné místo ve studijních plánech škol (SŠ i VŠ). Základním stavebním kamenem výuky je již zmíněný jazyk SQL, který je i přes svou poměrně jednoduchou strukturu a jasnou logiku velmi komplexním nástrojem, v kterém je možné jeden problém řešit několika různými způsoby. Jinak řečeno, jedno zadání může být úspěšně vyřešeno pomocí různých syntaxí SQL dotazů.

V rámci výuky je proto nutné studentům alespoň některé z těchto způsobů předvést, aby získali přehled o vhodnosti či nevhodnosti použití konkrétních postupů v konkrétních situacích. Je tedy zřejmé, že je nutné pro výuku databázových předmětů vyčlenit značný časový prostor, který je vhodné efektivně využít.

Problémově orientovaná výuka ovšem klade nebývalý důraz na důkladnou přípravu ze strany vyučujícího. Řešené příklady by měly provést studenty všemi zákoutími probírané látky. Pro důkladné osvojení učiva je nezbytné, aby dle připravených zadání realizovali studenti

na každém cvičení větší počet příkladů - od jednoduchých až po komplexnější. Druhým úskalím těchto cvičení je pak samotná kontrola řešení jednotlivých příkladů. Článek se zabývá automatizací procesu kontroly řešení jednotlivých databázových úloh.

## 2 Výuka databázových předmětů

Jak již bylo nastíněno v úvodu, výuka databázových systémů se stala jedním z pilířů studijních oborů zaměřených na informační technologie. Důležité je nejen to, aby si studenti odnesli komplexní teoretické znalosti, ale aby především získali rutinu při řešení běžných databázových úloh. Tj. aby studenti byli schopni dané úlohy dekomponovat na lépe řešitelné množiny jednodušších úloh a z jejich výsledků sestavit výsledek celé úlohy. Toho nelze ovšem dosáhnout jen teoretickou výukou.

Model výuky databázových předmětů na Fakultě elektrotechniky a informatiky na Univerzitě Pardubice zahrnuje standardní teoretické přednášky doplněné praktickými cvičeními, která na přednášky tematicky navazují. Hodinová dotace cvičení se pohybuje v rozmezí dvou až tří hodin týdně. Základní výuka databázových předmětů zaměřená na jazyk SQL a databázové modelování je rozdělena do dvou předmětů vyučovaných v navazujících semestrech během bakalářského studia. V navazujícím magisterském studiu pak na tuto výuku navazují další specializované databázové předměty.

Pro získání praktických dovedností při konstrukci příkazů v jazyce SQL je nezbytné úspěšně vyřešit dostatečné množství konkrétních zadání. Velice důležitým aspektem je zajištění zpětné vazby studentům, zda jejich řešení praktických úloh splňují či nesplňují zadání. Bez této zpětné vazby nemají ani studenti ani vyučující dostatečné informace o úspěšnosti studentů a není tedy možné správně upravit obsah další výuky, tj. například diskutovat se studenty příklady, při nichž se dopouštějí systematických chyb a znovu vysvětlit určitou látku učiva.

Úlohy, jež vedou k formulaci SQL dotazu, jsou formulovány tak, aby výstupním objektem byl pohled. Pohledy velice efektně řeší problematiku uložení konkrétních dotazů. Není tedy potřeba ukládat dotazy v různých externích formách (soubory se skriptem apod.). Každý pohled má svůj jednoznačný název, přes který je pak snadno identifikovatelný přímo ve schématu studenta. Při vytvoření pohledu se navíc ukládá v systémovém katalogu databáze řada doplňujících informací, které se využívají i při následné kontrole. Kromě vlastní syntaxe, názvů a typů atributů jmenujme například datum a čas vytvoření pohledu i jeho poslední změny, platnost objektu a podobně.

Udělejme si orientační výpočet. Pokud by vyučující měl manuálně zkontrolovat práci cca 200 studentů představující 5 až 10 příkladů každý týden, zjistíme, že je to prakticky nemožné a potřeboval by na to několik dní. Proto se doposud práce studentů kontrolovala spíše namátkově nebo frontálně během následujícího cvičení.

Naší primární snahou bylo proto hledání takového řešení, které by vyučujícím usnadnilo kontrolu praktické činnosti studentů během semestru, zápočtových příkladů a během zkoušek a studentům nabídlo okamžitou zpětnou vazbu o výsledku jejich práce. Čas, který vyučující ušetří, tak může být věnován přípravě nových praktických úloh - a to nejen zadání příkladů, ale také dat ve vzorových tabulkách, nad nimiž studenti dané SQL dotazy vytváří. Pro následující kontrolu práce studentů je příprava vhodných vzorových dat základním předpokladem pro rozlišení správných řešení od řešení, která sice v některých situacích poskytují shodné výsledky, nicméně v obecných případech by poskytovala výsledky rozdílné.

Nejdříve jsme pro kontrolu řešení používali ad-hoc tvořených databázových dotazů, které sice usnadnily kontrolu příkladů, ale vlastní tvorba těchto dotazů zabrala poměrně velké množství času. Navíc tyto dotazy nebyly opakovaně použitelné a výsledky kontrol se nearchivovaly. Nicméně tato fáze nám pomohla definovat konkrétní požadavky na vytvářenou aplikaci.

Pro studenty přináší daná aplikace řadu výhod. Prakticky okamžitě vědí, zda zadaný příklad vyřešili korektně či nikoliv. Tím pro ně odpadá nutnost na výsledek daného příkladu se neustále dotazovat či čekat například na konec cvičení, kdy se výsledky jednotlivých příkladů probírají. Systém pro kontrolu je navíc pro studenty dostupný 24 hodin denně, takže mohou příklady zhotovovat i z domova, opět s prakticky okamžitou zpětnou vazbou o úspěšnosti svého počínání.

Samozřejmě použití výše uvedeného softwarového řešení nezbavuje lektora povinnosti kontroly úplně. Například při cvičeních průběžně sleduje výsledky práce studentů při řešení jednotlivých úloh a případně usměrňuje jejich

činnost pro zdárné vyřešení zadaného příkladu. Problémové příklady zařazuje znovu do následujících cvičení pro jejich řádné procvičení.

## 3 Databázový systém

Podstatným elementem výuky je databázový systém. Pro výuku je na Fakultě elektrotechniky využíván databázový systém Oracle ve verzi 11g Release 2, jeden z nejpoužívanějších profesionálních databázových systémů. Databázový systém slouží nejen jako nástroj pro práci studentů, ale slouží i samotné aplikaci k ukládání dat a jsou v něm vytvořeny i všechny procedury a funkce používané ve vlastní kontrole.

Každý ze studentů má v rámci databázového systému svůj vlastní učet (v konvencích databáze Oracle nazývaný schéma). V tomto schématu student realizuje všechny zadané úkoly a archivuje je zde ve formě databázových objektů.

Mimo studentských schémat se v databázi nachází schémata, která můžeme označit jako referenční a schémata se vzorovými daty. Do referenčních schémat ukládají lektoři správná řešení praktických úloh (v podobě shodných databázových objektů, které mají studenti vytvořit). K referenčním schématům nemají studenti žádná práva, nemohou tedy dané objekty vidět. Databázové objekty v referenčních schématech jsou následně kontrolovány proti databázovým objektům vytvořených studenty. Ty budeme nadále nazývat jako testované.

Ve vzorových schématech se pak nacházejí podkladová data pro řešené úlohy. Dále je vzorovém schématu uložena tabulka, v níž je popis vzorového schématu a diagram fyzického databázového modelu.

## 4 Aplikace DB-CHECK

V současnosti je aplikace určena pro databázový systém Oracle (konkrétně Oracle 11g) a je schopna kontrolovat čtyři základní typy databázových objektů: tabulky, pohledy, funkce, procedury.

Pro výuku databázových předmětů během bakalářského studia je tento výčet dostačující. Pro magisterské předměty bude nutné aplikaci rozšířit i o kontrolu objektových typů, balíčků či XML datových typů a podobně.

Aplikace se skládá z lektorské a studentské části. Lektoři mají možnost vytvářet nové sady příkladů (takzvané šablony), vytvářet nová cvičení či prohlížet výsledky práce jednotlivých studentů v několika různých komplexních přehledech. Lektor může definovat v systému nové předměty, cvičení, zkouškové či zápočtové termíny. K tomu všemu může využít schopnosti aplikace synchronizovat seznamy předmětů, studentů a skupin s univerzitním informačním systémem STAG. To umožňuje lektorům pracovat vždy s aktuálním seznamem studentů na daném cvičení, zápočtu či zkoušce.

Studentská část slouží studentům ke kontrolám jejich vlastní práce. Student vidí pouze hodnocení svých příkladů. V detailu každého příkladu může například zjistit, v jakých parametrech se jeho řešení liší od řešení referenčního. Může tak například snadno odhalit, že se chyba nachází v definici objektu nebo že objekt nevrací správná data. U všech příkladů je možné samozřejmě nastavit, od kdy mohou studenti vidět hodnocení své práce (například až po zkoušce, nikoli během ní, byť vyučující tyto výsledky samozřejmě vidí i v průběhu zkoušky).

## 5 Princip práce s aplikací DB-CHECK

Základní příprava cvičení za využití aplikace DB-CHECK má přibližně následující podobu:

**1. Definice příkladů –** lektor definuje zadání příkladů pro cvičení. Příklady typicky obsahují tematiku probíranou na předcházející přednášce. Zároveň objekty definované těmito příklady vytvoří v referenčních schématech (budou využity jako referenční objekty při pozdější automatické kontrole).

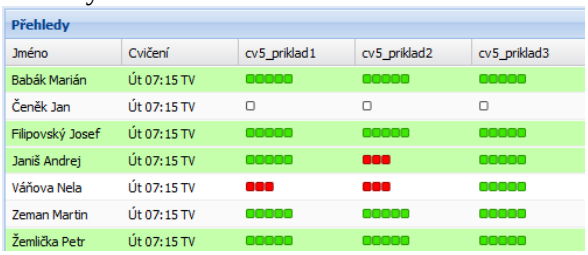
**2. Zadání šablony cvičení –** lektor vytvoří v aplikaci tzv. šablonu cvičení. Do šablony následně definuje jednotlivé úlohy pro konkrétní příklady. V definici každé úlohy nastaví odkaz na referenční databázový objekt, který bude pro danou úlohu sloužit jako kontrolní. Texty zadání jednotlivých úloh jsou dostupné studentům, a proto mohou sloužit přímo jako forma zadání těchto příkladů pro konkrétní rozvrhovou akci (např. cvičení, zápočtový test, zkouška) bez nutnosti vytváření dalších dokumentů se zadáním.

**3. Realizace cvičení –** lektor připojí šablonu ke konkrétnímu cvičení - tedy ke konkrétní studijní skupině a pro konkrétní termín konání cvičení. Od této chvíle se začnou kontrolovat příklady řešené studenty a výsledky plnění daných úloh budou dostupné lektorům i studentům (není-li uvedeno jinak) daného cvičení.

**4. Kontrola příkladů** – jednotlivé příklady studentů jsou průběžně automaticky kontrolovány kdykoliv student vytvoří nebo modifikuje databázový objekt, jehož název a typ odpovídají definovaným příkladům. Databázový systém obsahuje trigger, který reaguje na všechny DDL příkazy a pokud se týkají objektů, které mají být kontrolovány, požadavky jsou zařazeny do fronty a průběžně jsou vytvořené objekty kontrolovány. Lektor má možnost spustit kontrolu i manuálně přímo z aplikace, přičemž může určit její rozsah.

## 6 Uživatelské rozhraní aplikace DB-CHECK

Uživatelské rozhraní je vyvíjeno s ohledem na jednoduchost a přehlednost. Na obrázku č. 1 se nachází základní přehled plnění úloh studenty. Na obrázku č. 2 je zobrazen navigační strom, který slouží jako rozcestník do jednotlivých předmětů definovaných v systému a jejich rozvrhových akcí.


Přehledy

| Jméno | Cvičení | cv5_priklad1 | cv5_priklad2 | cv5_priklad3 |
|---|---|---|---|---|
| Babák Marián | Út 07:15 TV | ■■■■■ | ■■■■■ | ■■■■■ |
| Čeněk Jan | Út 07:15 TV | □ | □ | □ |
| Filipovský Josef | Út 07:15 TV | ■■■■■ | ■■■■■ | ■■■■■ |
| Janiš Andrej | Út 07:15 TV | ■■■■■ | ■■■ | ■■■■■ |
| Váňova Nela | Út 07:15 TV | ■■■ | ■■■ | ■■■■■ |
| Zeman Martin | Út 07:15 TV | ■■■■■ | ■■■■■ | ■■■■■ |
| Žemlička Petr | Út 07:15 TV | ■■■■■ | ■■■■■ | ■■■■■ |

***Obr. 1:*** *Základní přehled.*

Každý z předmětů vždy obsahuje položky pro definici šablon, cvičení, zkoušek a zápočtových úloh. V definici šablon se nacházejí předpřipravené sady příkladů, které lze následně přiřadit k jednotlivým termínům cvičení, zkoušek a zápočtů.

Aplikace DB-CHECK obsahuje také rozhraní na univerzitní informační systém STAG, které umožňuje:

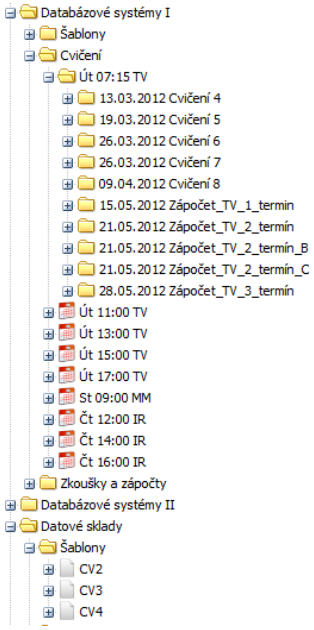
- vytvořit předmět a studijní skupiny pro jednotlivá cvičení,
- načíst zápočtové a zkouškové termíny a na ně přihlášené studenty.

Díky tomu je možné udržovat seznamy studentů na dané rozvrhové akci vždy aktuální.

V pravé části aplikace nazvané „Přehledy“ se nachází přehledová tabulka. V tabulce jsou přehledně graficky zobrazena plnění jednotlivých úloh konkrétními studenty vybrané rozvrhové akce. Možnosti zobrazení jsou velmi variabilní a zobrazení je ovlivněno vybranou položkou ve stromě navigátoru, například:

- přehled plnění příkladů daného cvičení všemi studenty předmětu nebo studenty konkrétního cvičení,
- přehled plnění příkladů všech cvičení všemi studenty předmětu nebo jen studenty zvolené studijní skupiny, zobrazení termínu zkoušky či zápočtu spolu se seznamem studentů a výsledky kontrol řešených úloh.

Pro lektory je velmi důležitou funkcionalitou aplikace možnost definování vlastních šablon. Z pohledu aplikace slouží šablona jako předloha pro realizaci konkrétní rozvrhové akce (cvičení, zápočtového termínu, zkoušky). Každá šablona definuje řadu příkladů specifických pro dané cvičení. Každou takto vytvořenou šablonu je možné v rámci daného předmětu připojit k libovolnému cvičení či zkoušce.




***Obr. 2:*** *Navigátor sloužící výběr konkrétní cvičení/zkoušky v daném předmětu.*

Podstatnou součástí šablony jsou úlohy. Z pohledu cvičení odpovídají úlohy jednotlivým příkladům. Každá takto definovaná úloha je specifická svým jménem a především svými parametry. U úloh se definují parametry jako limit splnění (procentuální hranice akceptace dané úlohy), typ databázového objektu (pohled, funkce, procedura, …) a umístění referenčního databázového objektu v rámci databázových schémat. U funkcí a procedur se ještě navíc definují tzv. testovací sady parametrů, se kterými se studenty vytvořené funkce a procedury postupně vykonávají, a jejich výstupy se pak porovnají s odpovídajícími výstupy referenčních funkcí či procedur.

Přehledová tabulka se výborně hodí pro rychlý přehled o dění v aktuálním cvičení (nebo zkoušce). Snadno se z tabulky určí, kteří studenti

úlohy plní a kteří ještě ne. Dá se například i vysledovat trend toho, které příklady se studentům daří řešit a které naopak ne. Na druhou stranu, vypovídající hodnota přehledové tabulky je omezená. Dozvíme se sice, že daný student má vybranou úlohu špatně, ale už se nedozvíme proč.

Proto aplikace umožňuje zobrazení detailu kontroly každé úlohy (viz obr. 6). Informace poskytované detailem se mírně liší v závislosti na typu kontrolovaného databázového objektu. Například u pohledů se kontroluje shoda řádků vracených referenčním a testovaným pohledem, kdežto u funkcí a procedur se testují shody výstupů. Detail kontroly dále poskytuje tyto informace:

- zda je objekt ve validním stavu,
- datum a čas vytvoření a poslední modifikace objektu (barevně odlišuje příklady, kdy poslední modifikace proběhla až po požadovaném termínu),
- počet shodných názvů a datových typů atributů (u pohledů a tabulek) či shodných parametrů (u procedur a funkcí),
- počet požadovaných (dle referenčního objektu) / skutečných (u testovaného objektu) / shodných řádků v pohledech a tabulkách,
- datum a čas poslední kontroly testovaného databázového objektu.
- odkazy vedoucí na zobrazení dat z referenčních objektů a dat z testovaných objektů vytvořených studenty..- Ve výpise jsou barevně odlišeny řádky a sloupce, kde se data neshodují s testovaným (či referenčním) objektem,
- odkazy pro zobrazení zdrojových kódů testovaných i referenčních objektů.

O viditelnosti dat a zdrojových kódů referenčních objektů pro studenty vždy rozhoduje daný lektor při vytváření úlohy v šabloně. Typicky jsou zpřístupněna data referenčního objektu, zdrojové kódy však nikoliv.

## 7 Praktická ukázka aplikace DB-CHECK

Uvažujme následující modelovou úlohu. Lektor definuje jednoduchý příklad, který vede na využití vnějšího spojení tabulek. Zároveň lektor definuje nové názvy sloupců (tzv. alias), které se liší od názvů ve vzorových tabulkách. Lektor tedy kromě zadání vytvoří v databázi i samotný referenční objekt (obr. 3) a zadá příklad do aplikace DB-CHECK, v tomto případě pohled. Pro vytvoření objektu lektor využije následujícího příkazu:

```
CREATE OR REPLACE VIEW priklad_1 AS
SELECT
  prijmeni, oddeleni_nazev AS oddeleni
FROM
  A_HR.zamestnanci
  LEFT JOIN A_HR.oddeleni ON
    zamestnanci.oddeleni_id = oddeleni.oddeleni_id
```

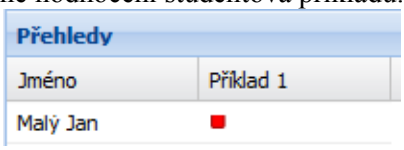
***Obr. 3:*** *Definice referenčního objektu (vytváří lektor).*

Student po přečtení zadání vytvoří své vlastní řešení daného příkladu, viz např. syntaxe dle obr. 4. Jakmile příkaz provede, aplikace následně automaticky provede kontrolu jím vytvořeného objektu proti objektu, který je pro tento příklad definován jako referenční. Pro zajištění kontroly je nezbytné dodržet název testovaného objektu dle zadání úlohy.

```
CREATE OR REPLACE VIEW priklad_1 AS
SELECT
  prijmeni, oddeleni_nazev AS oddelenni
FROM
  A_HR.zamestnanci
  JOIN A_HR.oddeleni ON
    zamestnanci.oddeleni_id = oddeleni.oddeleni_id
```

***Obr. 4:*** *Definice studentova (testovaného) objektu.*

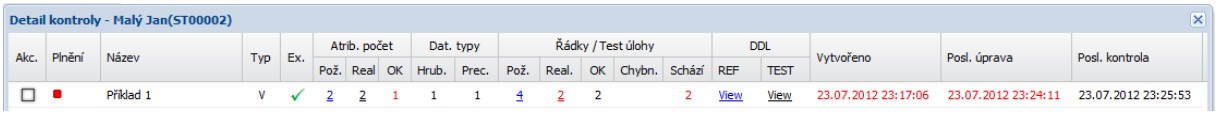
Z kódu studentova objektu je patrné, že se dopustil dvou chyb. Za prvé místo vnějšího spojení využil spojení vnitřní a za druhé chybně přejmenoval jeden ze sloupců. Tyto, ač na první pohled drobné chyby, mají následně zásadní vliv na výsledné hodnocení studentova příkladu.




***Obr. 5:*** *Výsledné hodnocení studentova řešení daného příkladu.*

Student obdržel velmi nízké hodnocení (obr. 5). Nyní mají lektor i student možnost zobrazit detail hodnocení (obr. 6) dané úlohy a zjistit, co student konkrétně dělá špatně.

Z detailu je na první pohled zřejmé, že první chybou, která snižuje celé hodnocení, je nesprávně definovaný název jednoho atributu objektu (u pohledu se atributy myslí sloupce). Vidíme, že daný příklad vyžaduje dva atributy. Student sice dva atributy definoval, nicméně správně je nazvaný pouze jeden. Hodnoty ve sloupcích „Pož.“ a „Real“ jsou hypertextovými odkazy, kterou vedou na tabulku detailů atributů referenčního objektu (obr. 7) a na detailů atributů testovaného studentova objektu (obr. 8).



Detail kontroly - Malý Jan(ST00002)

| Akc. | Plnění | Název | Typ | Ex. | Atrib. počet | | | Dat. typy | | Řádky / Test úlohy | | | | | DDL | | Vytvořeno | Posl. úprava | Posl. kontrola |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | Pož. | Real | OK | Hrub. | Prec. | Pož. | Real. | OK | Chybn. | Schází | REF | TEST | | | |
| ☐ | ■ | Příklad 1 | V | ✓ | 2 | 2 | 1 | 1 | 1 | 4 | 2 | 2 | | 2 | View | View | 23.07.2012 23:17:06 | 23.07.2012 23:24:11 | 23.07.2012 23:25:53 |

***Obr. 6:*** *Detail kontroly daného studentova příkladu.*

STR - "Příklad 1"

| Pořadí | Atribut | Datový typ | Délka | Přesnost | Char/Byte |
|---|---|---|---|---|---|
| 1 | PRIJMENI | VARCHAR2 | 25 | | B |
| 2 | ODDELENI | VARCHAR2 | 30 | | B |

***Obr. 7:*** *Seznam parametrů referenčního objektu. Žlutě zvýrazněný řádek indikuje, že student se v tomto parametru odlišuje.*

STR - "Příklad 1" - Malý Jan (ST00002)

| Pořadí | Atribut | Datový typ | Délka | Přesnost | Char/Byte |
|---|---|---|---|---|---|
| 1 | PRIJMENI | VARCHAR2 | 25 | | B |
| 2 | ODDELENNI | VARCHAR2 | 30 | | B |

***Obr. 8:*** *Seznam parametrů studentova objektu. Červený řádek indikuje, že daný parametr není správně definován.*

Z detailu kontroly je dále zřejmé, že ani výstup studentova objektu není vzhledem k referenčnímu objektu v pořádku. V tomto případě můžeme vidět, že referenční objekt (v tomto případě pohled) vrací 4 řádky, kdežto studentův pohled vrací jen řádky dva (zřejmý důsledek použití nevhodného typu spojení tabulek). Sloupce „Pož.“ a „Real“ opět slouží jako hypertextové odkazy, které v tomto případě vedou na detail dat, jenž referenční (obr. 9) a studentův objekt (obr. 10) generují a umožňují jejich přímé porovnání.

DATA - "Příklad 1"

| Číslo řádku | PRIJMENI | ODDELENI |
|---|---|---|
| 1 | Gietz | Accounting |
| 2 | Higgins | Accounting |
| 3 | Grant | |
| 4 | Philtanker | |

***Obr. 9:*** *Přehled dat generovaných referenčním objektem. Žlutě označené řádky značí chybějící nebo špatné řádky ve studentově objektu.*

Dále může detail poskytnout lektorovi zdrojový kód studentova databázového objektu. To je velmi užitečné v případě hledání chyb v řešení studentů. Někdy může být i nápomocna při odhalování, zda studenti nekopírují jen řešení svých kolegů. Užitečné jsou i informace o datu prvního vytvoření daného objektu a datu jeho poslední modifikace.

DATA - "Příklad 1" - Malý Jan (ST00002)

| Číslo řádku | PRIJMENI | ODDELENNI |
|---|---|---|
| 2 | Gietz | Accounting |
| 1 | Higgins | Accounting |

***Obr. 10:*** *Data generovaná studentovým objektem.*

## 8 Závěr

Aplikace DB-CHECK velmi zefektivnila výuku databázových systému na Fakultě elektrotechniky a informatiky Univerzity Pardubice. Validace příkladů je nyní velmi rychlá a spolehlivá. Okamžitě je možné studentovi ukázat, kde je chyba, neboť aplikace umožňuje rychlý náhled na studentovy výsledky i kód objektu. Tím se ušetří velmi mnoho času, který bylo nutné kdysi trávit vyhledáváním objektů studentů a porovnáváním jejich výstupu s výstupy referenčních objektů. Tento čas je nyní možné věnovat samotné výuce a tak proniknout hlouběji do podstaty probírané látky.

I pro samotné studenty je aplikace velmi přínosná. Okamžitě vědí, zda mají daný příklad úspěšně vyřešen. Pokud student příklad v pořádku nemá, mohou mu podrobné statistiky o jeho úloze napovědět, kde chybuje. Díky tomu, že studenti mají možnost validace svých výsledků a vědí, kdy mají příklad splněn, rapidně vzrostl počet příkladů, které jsou studenti schopni úspěšně dokončit.

Aplikace se také osvědčila při zápočtových testech a při praktické části zkoušek. Dříve si musel lektor pomáhat různými dotazy do systémového katalogu či dotazovat přímo studentovy databázové objekty. Nyní má výsledky kontrol okamžitě k dispozici a může se věnovat přímo rozboru studentova řešení.

## 9 Literatura

[1] GROFF, James R a Paul N WEINBERG. SQL: kompletní průvodce. Vyd. 1. Brno: CP Books, 2005, 936 s. ISBN 80-251-0369-2.
[2] VAŇKOVÁ, Jana a Michal ČERNÝ. Několik podnětů k výuce databází. [online]. 2011 [cit. 2012-08-02]. Dostupné z: http://clanky.rvp.cz/clanek/k/g/13993/NEKOLIK-PODNETU-K%C2%A0VYUCE-DATABAZI.html/

**Ing. Tomáš Váňa**
**RNDr. David Žák, Ph.D.**
**Ing. Jiří Zechmeister**
**Ing. Jiří Lebduška**
**Katedra informačních technologií**
**Fakulta elektrotechniky a informatiky UCPE**
**Studentská 95**
**532 10, Pardubice, ČR**
**Tel: +420 466 037 060**
**E-mail: tomas.vana@student.upce.cz,**
**david.zak@upce.cz,**
**jiri.zechmeister@student.upce.cz,**
**jiri.lebduska@student.upce.cz**
**Www pracoviště: www.upce.cz/fei/kit.html**