Popis funkcí databáze Oracle interMedia pro práci s obrázky a vytvoření vzorových příkladů pro cvičení k předmětu Teorie zpracování dat

Function description of Oracle interMedia database for work with pictures and creating examples for subject Theory of data processing

David Filípek

Bakalářská práce
2010

Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky

Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky
akademický rok: 2009/2010

ZADÁNÍ BAKALÁŘSKÉ PRÁCE
(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení:	**David FILÍPEK**
Osobní číslo:	**A07544**
Studijní program:	**B 3902 Inženýrská informatika**
Studijní obor:	**Informační a řídicí technologie**
Téma práce:	**Popis funkcí databáze Oracle interMedia pro práci s obrázky a vytvoření vzorových příkladů pro výuku cvičení k předmětu Teorie zpracování dat.**

Zásady pro vypracování:

1. **Nastudujte funkce Oracle interMedia podle originální anglické dokumentace.**
2. **Vymyslete vlastní příklady použití funkcí Oracle interMedia.**
3. **Vyzkoušejte funkčnost příkladů na školním serveru s databází Oracle.**
4. **Vytvořte českou uživatelskou příručku, která bude obsahovat popis funkcí Oracle interMedia a jejich ukázky použití na Vámi vytvořených příkladech.**

Rozsah bakalářské práce:

Rozsah příloh:

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

Seznam odborné literatury:

1. **Loney, Kevin., Bryla, Bob. Mistrovství v Oracle Database 10g. Praha : Computer Press, 2006 704 s. ISBN: 80-251-1277-2**
2. **Kyte, Thomas. Oracle : Návrh a tvorba aplikací. Praha : Computer Press, 2005. 696 s. ISBN: 80-251-0569-5**
3. **Introduction to Oracle interMedia [online]. c2005 [cit. 2009-12-27]. Dostupný z WWW https://students.kiv.zcu.cz/doc/oracle/appdev.102/b14302/ch_intr.htm**
4. **Oracle Database Documentation Library : Oracle interMedia Reference [online]. c2009[cit.2009-12-27]. Dostupný z WWW http://www.oracle.com/pls/db102/portal.portal_db?selected=7**
5. **Oracle Database Documentation Library : Oracle interMedia Users Guide [online]. c2009[cit. 2009-12-27]. Dostupný z WWW http://www.oracle.com/pls/db102/portal.portal_db?selected=7**

Vedoucí bakalářské práce: **Ing. Kateřina Ježková**
Ústav automatizace a řídicí techniky

Datum zadání bakalářské práce: **5. března 2010**

Termín odevzdání bakalářské práce: **1. června 2010**

Ve Zlíně dne 5. března 2010

L.S.

UNIVERZITA TOMÁŠE BATI VE ZLÍNĚ
Fakulta aplikované informatiky

prof. Ing. Vladimír Vašek, CSc.
děkan

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

ABSTRAKT

Tato práce je příspěvkem k podpoře vyučování předmětu Teorie zpracování dat. Hlavním cílem je popis metod a funkcí, kterých lze použít při práci s databází Oracle interMedia. Postupně je vysvětleno inicializování objektových typů, ukládání různých typů dat a široké možnosti při práci s těmito daty. Ve vývojovém prostředí SQL Developer byly vytvořeny a popsány vzorové příklady, které slouží pro jednodušší pochopení možností práce s databází.

Klíčová slova: Objektový typ, datový typ, metoda, objekt

ABSTRACT

This thesis is focused to support the subject „Theory of data processing“. The main aim is to introduce methods and functions that can be used for Oracle interMedia database. Initialization of object types, storage of various types of data and numerous options for work with these data are further described here. The development stage SQL Developer produces some user-friendly sample examples.

Keywords: Object type, data type, method, object

Chtěl bych zde uvést poděkování vedoucí mé práce Ing. Kateřině Ježkové za poskytnutí odborných připomínek a návrhů jak postupovat pří řešení dané práce.

Prohlašuji, že

- beru na vědomí, že odevzdáním bakalářské práce souhlasím se zveřejněním své práce podle zákona č. 111/1998 Sb. o vysokých školách a o změně a doplnění dalších zákonů (zákon o vysokých školách), ve znění pozdějších právních předpisů, bez ohledu na výsledek obhajoby;
- beru na vědomí, že bakalářská práce bude uložena v elektronické podobě v univerzitním informačním systému dostupná k prezenčnímu nahlédnutí, že jeden výtisk bakalářské práce bude uložen v příruční knihovně Fakulty aplikované informatiky Univerzity Tomáše Bati ve Zlíně a jeden výtisk bude uložen u vedoucího práce;
- byl/a jsem seznámen/a s tím, že na moji bakalářskou práci se plně vztahuje zákon č. 121/2000 Sb. o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon) ve znění pozdějších právních předpisů, zejm. § 35 odst. 3;
- beru na vědomí, že podle § 60 odst. 1 autorského zákona má UTB ve Zlíně právo na uzavření licenční smlouvy o užití školního díla v rozsahu § 12 odst. 4 autorského zákona;
- beru na vědomí, že podle § 60 odst. 2 a 3 autorského zákona mohu užít své dílo – bakalářskou práci nebo poskytnout licenci k jejímu využití jen s předchozím písemným souhlasem Univerzity Tomáše Bati ve Zlíně, která je oprávněna v takovém případě ode mne požadovat přiměřený příspěvek na úhradu nákladů, které byly Univerzitou Tomáše Bati ve Zlíně na vytvoření díla vynaloženy (až do jejich skutečné výše);
- beru na vědomí, že pokud bylo k vypracování bakalářské práce využito softwaru poskytnutého Univerzitou Tomáše Bati ve Zlíně nebo jinými subjekty pouze ke studijním a výzkumným účelům (tedy pouze k nekomerčnímu využití), nelze výsledky bakalářské práce využít ke komerčním účelům;
- beru na vědomí, že pokud je výstupem bakalářské práce jakýkoliv softwarový produkt, považují se za součást práce rovněž i zdrojové kódy, popř. soubory, ze kterých se projekt skládá. Neodevzdání této součásti může být důvodem k neobhájení práce.

Prohlašuji,

- že jsem na bakalářské práci pracoval samostatně a použitou literaturu jsem citoval. V případě publikace výsledků budu uveden jako spoluautor.
- že odevzdaná verze bakalářské práce a verze elektronická nahraná do IS/STAG jsou totožné.

Ve Zlíně …….

podpis diplomanta

OBSAH

ÚVOD

Tato bakalářská práce slouží k usnadnění a lepšímu pochopení správy databáze Oracle interMedia.Jedná se o databázový systém, který pracuje s multimediálními daty jako jsou obrázky, audio nahrávky a video záznamy.Databáze slouží pro usnadnění správy dat, kterých je v dnešní době všude velké množství a je potřeba s nimi pracovat efektivně. Velmi důležitý je celkový návrh databáze. Od rozdělení položek do jednotlivých tabulek, aby se některé položky nezadávaly vícekrát, po přiřazení datových nebo objektových typů pro ukládaná data a určení omezení, které je pro daná data potřebné. To může být například nutnost některé položky vyplnit vždy a některé ne. Dobře navržená databáze tak umožňuje jednoduché vyhledávání konkrétní informace ze všech dat, která jsou v databázi uložena. Další nespornou výhodou je rychlost za jakou lze zjistit potřebnou informaci, a to i v případě, že se nejedná o příliš rozsáhlou databázi.

V jednotlivých kapitolách jsou popsány různé metody a funkce, které umožňují rozsáhlou práci s multimediálními daty a jejich možnosti použití. Jednotlivé metody jsou popsány srozumitelně a ve snaze, aby jejich pochopení bylo co nejjednodušší.Jsou metody, které jsou jednoduché a dají se použít intuitivně, aniž by byla potřeba se jimi nějak více zabývat. Jsou ovšem i metody, které mají možnosti použití velmi rozsáhlé a pro jejich používání je již dobré vědět, co všechno lze s danou metodou provést, aby se dosáhlo požadovaného výsledku. V této práci se jedná pouze o základní popis možností, kterých lze s databázovým systémem interMedia dosáhnout.

Databáze Oracle interMedia je velice rozsáhlý systém, který nabízí spousty možností, kterých je možné při ukládání, vyhledávání i úpravě všech druhů multimediálních dat dosáhnout. Jedná se o systém, který již využívá možností objektového přístupu k databázi. Objektový přístup se nepoužívá jen při práci s databázemi, ale využívá se i v jiných technologiích a oborech, například objektové programování je dnes nejrozšířenější způsob programování aplikací.

Všechny příklady, které jsou uvedenév příloze bakalářské práce, byly vytvářeny ve vývojovém prostředí SQL Developer od firmy Oracle. Jedná se o jednoduché příklady, které jsou určené pro snazší pochopení jednotlivých metod, které jsou součástí databáze Oracle interMedia.

I. TEORETICKÁ ČÁST

1 DATABÁZOVÝ SYSTÉM ORACLE

Oracle je multiplatformní databázový systém s pokročilými možnostmi zpracování dat a vysokým výkonem. Databázový systém je vyvíjen firmou Oracle Corporation.

1.1 Historie Oracle

V roce 1977 založili Larry Ellison, Bob Miner a Ed Oates firmu SDL (Software Development Laboratory), která se zabývala vývojem RDBS (Relational Database Managment Systém).

V roce 1982 verze Oracle V3. pro sálové počítače, minipočítače i PC. Podporuje transakční zpracování.

Společenost se přejmenovala na Oracle Corporation v roce 1983. V té době vznikaly postupně verze databáze pod názvy Oracle V1 až po verzi Oracle V6. V roce 1989 vznikla verze s označením Oracle 6.2.

V roce 1994 vznikla verze Oracle 7 pro platformu PC.

Ve verzi Oracle 8, která vznikla v roce 1997, dochází k podpoře rozsáhlých databází. Vznikly nové datové typy pro ukládání obrazových a multimediálních dat (BLOB, CLOB, BFILE). Podpora objektově orientovaných technologií.

Ve verzi Oracle 8i dochází k integraci jazyka Java a dochází k orientaci na internetové technologie.

V současné době jsou používány verze Oracle 10g Relase 2 a verze Oracle 11g.

2 ORACLE INTER MEDIA

2.1 Popis databáze

Databáze Oracle interMedia (dále jen interMedia) je určena pro ukládání, k opětovnému získání a pro správu multimediálních dat. Při práci s multimediálními daty se používá 4 objektových typů (ORDAudio, ORDDoc, ORDVideo a ORDImage), které mají na starosti nejen vlastní uložení dat, ale poskytují i metody pro přístup k metadatům daného objektu. InterMedia obsahuje i funkce pro manipulaci s obsahem dat, kde lze např. změnit typ komprese, velikost apod. Je možné i vytvářet náhledy obrázku.

Databáze neumí data zobrazit, ale umí porovnávat jejich obsah. Pomocí jazyka SQL nebo jazyka PL/SQL (Procedural Language / Structered Query Language), který je procedurální nadstavbou jazyka SQL. V něm lze vytvářet uložené procedury a funkce, programové balíky (packages), triggery nebo uživatelsky definované datový typy.

InterMedia je rozšiřitelná databáze, která podporuje většinu známých formátů multimediálních dat, ale může se i rozšířit o další typy. To lze použít v případě vzniku nových digitálních kompresí a dekompresí.

2.2 Možnosti

InterMedia je přístupná k aplikacím přes relační a objektové rozhraní. Instance objektů obsahují atributy (vlastnosti), včetně metadat, media dat a metod.

- media data – audio, video, obrázek
- metadata – jsou informace o datech jako je délka (velikost) objektu, typ komprese nebo použitého formátu
- metody – procedury, které lze s objekty vykonávat (např. získat obsah)

2.3 Ukládání dat do databáze

Data mohou být ukládána do databáze jako datový typ interMedia nebo přímo jako soubor (pdf, doc apod.). V databázi jsou data uložena jako BLOB (binary large object) nebo BFILE.

BLOB

- data jsou uložena přímo v databázové tabulce a jsou pod transakční kontrolou
- soubory do 4 kB mohou být uloženy inline (přímo na řádku v dané tabulce), v případě větších souborů (až do 4 GB) je na řádek uložen pouze ukazatel na daný soubor a soubor je uložen do jiné tabulky

BFILE

- zde je v databázi uložen pouze ukazatel na daný soubor, který je uložen jako URL na http serveru nebo na uživatelem definovaném specializovaném datovém serveru
- data jsou uložena bez transakční kontroly, to znamená, že lze změnit obsah jejich dat nebo je vymazat aniž by se při tom aktualizovala databáze, to způsobí nesrovnalost mezi daty a ukazatelem na ně

2.4 Objektově relační technologie

Provádí podporu pro definici typů objektů, včetně dat spojených s objekty a operace (metody), které lze s nimi provádět. Objektové typy podporují běžné požadavky aplikací a mohou být rozšířené i na speciální požadavky aplikací.

3 OBJEKTOVÉ TYPY

3.1 Audio – ORDAudio

Datový typ ORDAudio slouží ke zpracování digitalizovaných zvukových nahrávek a k používání a vytváření audio aplikací nebo specializovaných ORDAudio objektů.

3.1.1 Digitalizované audio

Jedná se o zvuk zpracovaný do digitální podoby s charakteristikami jako je formát, typ kódování, četnost vzorkování, typ komprese a doba trvání zvukového záznamu. Velikost digitalizovaného audia směřuje k tomu, aby byla srovnatelná s tradičními počítačovými objekty jako jsou čísla a text. Proto se používá několik typů kódování, které sníží velikost dat na několik bytů.

3.1.2 Audio komponenty

Audio aplikace mohou obsahovat informace o datumu záznamu, popis nahrávky, jméno autora apod.

ORDAudio může ukládat a získávat audio data z jakéhokoliv z podporovaných formátů. (3GP, MPEG, WAV …).

3.2 Obrázky – ORDDImage

Používá se pro práci s digitálními obrázky. Existuje ještě i datový typ ORDImageSignature, který je ale podporován pouze v rozšířené verzi Enterprise Edition.

3.2.1 Digitalizované obrázky

ORDImage podporuje 2D statické obrázky uložené jako binární reprezentaci objektů reálného světa. Jednotlivé body v obrázku se nazývají pixely.

3.2.2 Součásti obrázků

Obrázky obsahují data a atributy, které popisují a charakterizují daný obrázek. Mohou obsahovat informace o jméně osoby na obrázku, popis obrázku, datum vzniku apod.

V ORDImage lze ukládat a znovu získat obrázková data jakéhokoliv formátu. ORDImage může zpracovávat a automaticky získávat vlastnosti obrázků z podporovaných formátů.

Pro ukládání dat je možné použít beztrátovou kompresi, to znamená, že po dekomprimaci dat je obrázek ve stejné kvalitě jako ten původní. Pří ztrátové kompresi dochází k částečné ztrátě dat, která je však pro lidské oko nepostřehnutelná. Výhodou je mnohem větší komprese a tím menší velikost dat.

3.2.3 Metadata v obrázcích

Metadata jsou data o datech a slouží k jednoduššímu vyhledávání. Ve verzi Oracle database 10g Relase 2 jsou metadata do databáze přídána. Umožňují číst a zapisovat metadata do obrázků. Jsou metadata technická (výška a šířka obrázku, počet pixelů…) nebo obsahová (popis obrázku, jméno autora, datum vytvoření…).

Od verze Oracle databaze 10g Relase 2 je do interMedia přidán také DICOM. Slouží ke správě dat v nemocnicích. DICOM objekty mohou ukládat data různých typů. Může to být např. informace o pacientovi, fotka, video atd. InterMedia umí v databázi rozpoznat DICOM objekty a získat tak potřebné informace.

3.2.4 Zpracování obrázků

V interMedia je také podporováno zpracování obrázků, jako je změna kódování, změna měřítka, oříznutí obrázku atd. Lze vytvořit i thumbnail (miniaturní náhled obrázku).

3.2.5 Objektový typ ORDImageSignature

V databázi Oracle interMedia je ještě používán objektový typ ORDImageSignature, který rozšiřuje možnosti práce s obrázky. Jedná se o tzv. podpis obrázku, pomocí kterého je možné porovnávat jednotlivé obrázky mezi sebou na základě shody vybraných parametrů.

3.3 Video – ORDVideo

Používá se pro práci s video objekty, jejich zpracování a uložení do databáze a možnost znovu získání z databáze.

3.3.1 Digitalizované video

Charakteristiky digitalizovaného videa jsou formáty, typy kódování, typy kompresí, velikosti rámců, rozlišení rámců, čas přehrávání, počet barev a přenosová rychlost.

3.3.2 Popis

Digiralizované video obsahuje data a atributy, které popisují a charakterizují konkrétní video data. Video aplikace mohou obsahovat informace například datum záznamu, jméno autora ad. ORDVideo může ukládat a zobrazovat data všech podporovaných formátů.

Podporované formáty

- Apple quick time 3.0
- Microsoft video for Windows (AVI)
- 3GP
- Real Networks Real Video
- MPEG (1, 2, 4)

Stejně jako u ostatních typů dat se používá kompresí, aby se zmenšílo množství dat, které je potřeba ukládat.

3.4 Heterogeneous (různorodá) data – ORDDoc

Tato kapitola, popisuje datový typ, který umí pracovat s různými typy multimediálních dat.

3.4.1 Digitalizovaná data

ORDDoc umí ukládat a spravovat všechny typy multimediálních dat. Zahrnuje audio, video i obrázková data. Na rozdíl od ostatních datových typů, kde každý datový typ potřebuje vlastní sloupec v tabulce databáze, ORDDoc umí ukládat všechny typy dat do jednoho sloupce. Toho je možné využít v případě, když je potřeba ukládat data a dopředu není jisté, zda to bude obrázek nebo například video ukázka.

3.4.2 Popis

Heterogeneous data mohou mít různé formáty, záleží na aplikaci, která data vytváří. ORDDoc se používá v aplikacích, které požadují ukládání typů dat ve stejném sloupci, takže lze vytvořit index metadat na všechny typy dat. Použitím tohoto indexu je možné vyhledávat různé typy dat. Tenhle způsob nelze použít, když jsou různé typy dat uloženy v různých typech objektů, v různých sloupcích tabulky.

II. PRAKTICKÁ ČÁST

4 PROSTŘEDKY PRO TVORBU PŘÍKLADŮ

Cílem při vytváření příkladů bylo navrhnout takové databázové dotazy a příkazy, které by jednoduše a přehledně popisovaly možnosti, kterých lze při používání databáze Oracle interMedia dosáhnout. Při vytváření příkladů je použito jazyka PL/SQL, vyvinutého firmou Oracle. Tento jazyk je rozšířením jazyka SQL. Jazyk PL/SQL doplňuje sílu SQL pro manipulaci s daty o sílu procedurálních jazyků. Lze používat SQL příkazy pro manipulaci s daty a navíc jde deklarovat konstanty a proměnné, definovat procedury a funkce a s využitím vyjímek je možné zpracovávat chyby, které vznikly za chodu programu. Jazyk PL/SQL se používá ve všech produktech systému Oracle (SQL*PLUS, Forms Builder…). Syntaxe jazyka vychází z programovacího jazyku Ada.

4.1 Vývojové prostředí

Pro práci s databází byl vybrán program Oracle SQL Developer. Je vyvíjen firmou Oracle a je ke stažení zdarma na stránkách Oracle. Jedná se o grafické prostředí, pomocí kterého je možné spravovat databáze. Je především určen pro spravování databází Oracle, ale je možné ho použít i pro práci s jinými databázemi (MySQL, Microsoft SQL Server ad.). Je to hlavně z důvodu možnosti importu dat z těchto databází. SQL Developer byl před verzí 1.0 znám jako Raptor.

4.1.1 Instalace

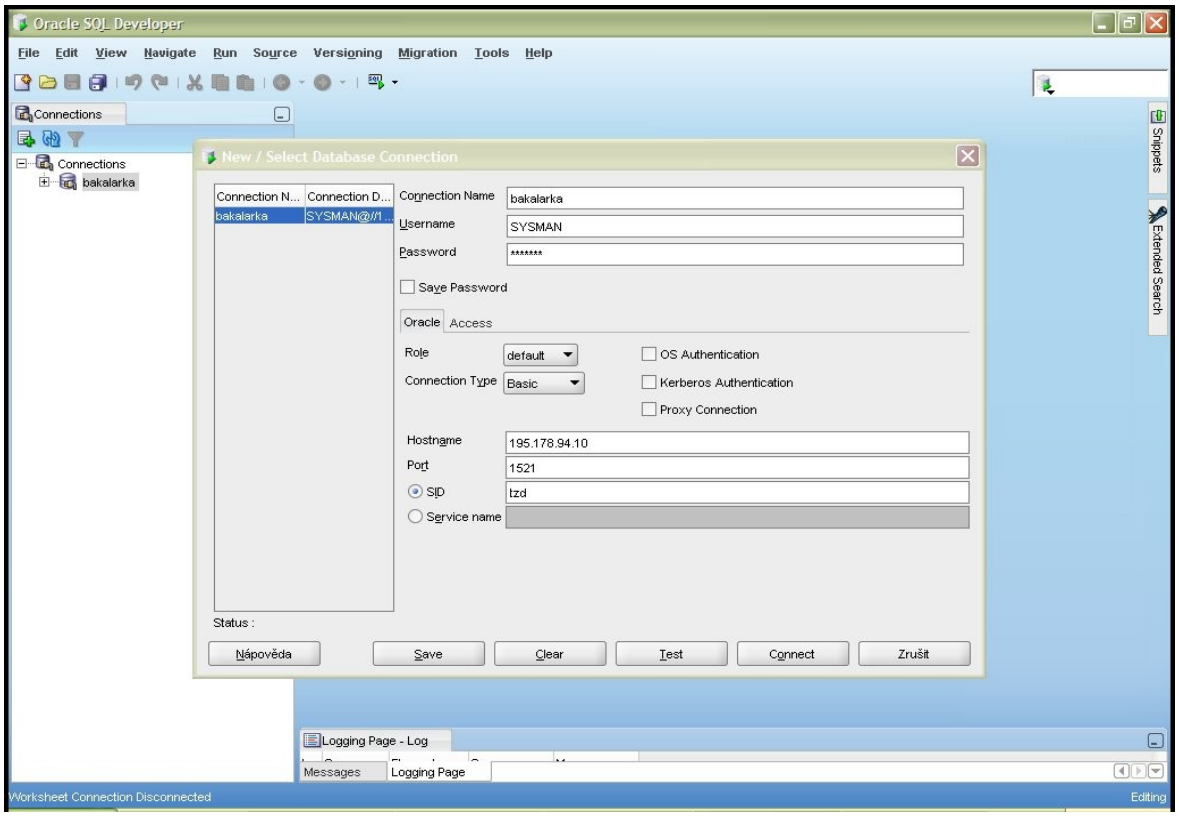
SQL Developer je program, který je napsán v jazyku Java, a proto se neinstaluje. Pouze se rozbalí stažený zip archiv. Při vytváření příkladů pro tuto bakalářskou práci byla použita verze SQL Developer 1.5.5. Je dostupný pouze v angličtině. Dále je nutné nainstalovat JDK (Java Development Kit) verze 1.5.0 nebo vyšší. Pro operační systémy Windows je možné JDK stáhnout společně se SQL Developer v jednom zip archivu. Pro operační systémy Linux a MacOS je potřeba si JDK stáhnout a nainstalovat samostatně. Vzhledem k tomu, že je SQL Developer napsán v jazyku Java je ještě potřeba mít nainstalované JRE (Java Runtime Environment).

4.1.2 Připojení

Pro vytváření příkladů k přiložené příručce bylo použito připojení na školní server, kde je nainstalována verze databáze Oracle 10g Release 2. Komunikace s databází probíhala přes nainstalovaného klienta, v tomto případě pomocí programu SQL Developer.

Po prvním spuštění je nabídnut import dat a nastavení z předchozích verzí (pokud jsou dostupné). Pro vytvoření nového připojení stačí kliknout pravým tlačítkem na Connections a vybrat položku New Connection. Otevře se nové okno kde je potřeba zadat položky:

- Connection Name – název připojení
- Username – uživatelské jméno
- Password – heslo (je možné ho trvale uložit)
- Hostname – adresa serveru pro připojení
- Port – číslo portu
- SID – jméno instance pro připojení



Obr. 1. Přihlašovací okno

Pro kontrolu zda je všechno zadáno v pořádku stačí použít tlačítko Test. Poté je možné zmáčknout tlačítko Connect pro připojení k databázi. Po úspěšném připojení se v záložce Connections rozbalí stromová struktura databáze (tabulky, indexy, adresáře …) a zobrazí se nová záložka (název připojení), kde je možné psát SQL dotazy. Ty je možné spouštět stiskem klávesy F9 nebo myší kliknutím na zelenou šipku.

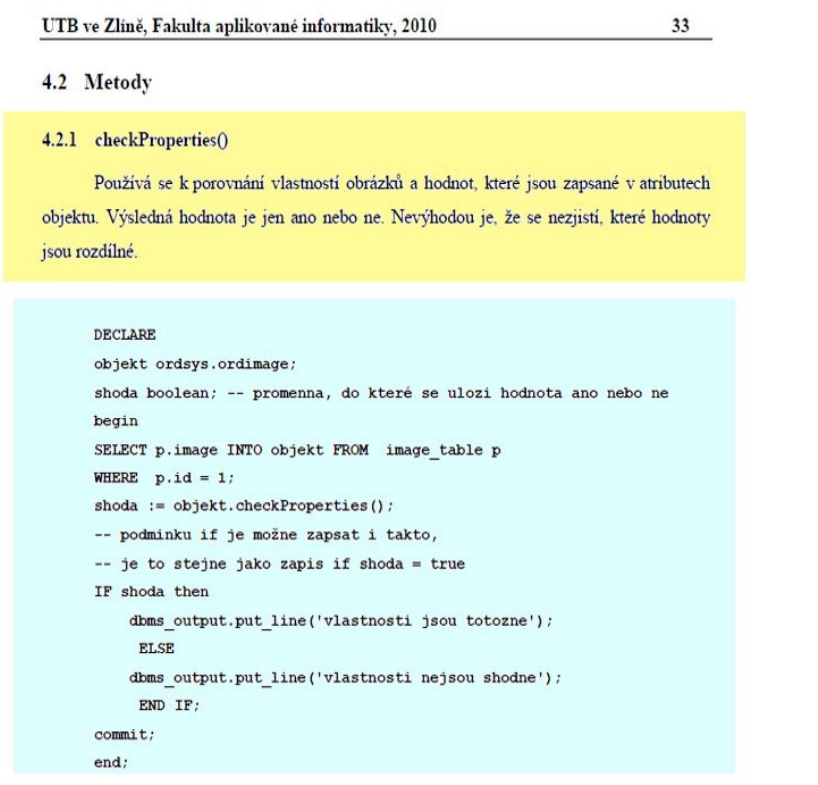
Výsledky dotazů se zobrazí ve spodní části okna v záložce Results. V případě cvičných příkladů vytvářených pro práci s multimediálními daty jsou výsledky zobrazovány pomocí příkazu *dbms_output.put_line(´´)*. Pro zobrazení výsledků tohoto příkazu je potřeba se přepnout ze záložky Results do záložky DBMS Output.

5 PŘÍKLADY A UŽIVATELSKÁ PŘÍRUČKA

V této kapitole jsou zahrnuty možnosti používání metod pro práci s objekty v databázi a popis tvorby příkladů, které jsou součástí přiložené uživatelské příručky. Jedná se o metody pomocí kterých lze zadávat do atributů objektu vlastnosti multimediálních dat i o metody, které z těchto atributů získávají informace pro vyhledávání požadovaných dat.

5.1 Uživatelská příručka

Příručka popisuje a vysvětluje použití jednotlivých metod, které lze použít při práci s databází Oracle interMedia. K metodám jsou připojeny ukázkové příklady, na kterých je ukázáno jak je možné danou metodu použít. Pro lepší pochopení použití některých metod, jsou v jednom příkladu uvedeny metody, které spolu souvisí a mohou se navzájem doplňovat. Do příkladů jsou vloženy komentáře i pomocné výpisy na obrazovku pro lepší přehlednost.



UTB ve Zlíně, Fakulta aplikované informatiky, 2010 33

4.2 Metody

4.2.1 checkProperties()

Používá se k porovnání vlastností obrázků a hodnot, které jsou zapsané v atributech objektu. Výsledná hodnota je jen ano nebo ne. Nevýhodou je, že se nezjistí, které hodnoty jsou rozdílné.

```
DECLARE
objekt ordsys.ordimage;
shoda boolean; -- promenna, do které se ulozi hodnota ano nebo ne
begin
SELECT p.image INTO objekt FROM  image_table p
WHERE  p.id = 1;
shoda := objekt.checkProperties();
-- podminku if je možne zapsat i takto,
-- je to stejne jako zapis if shoda = true
IF shoda then
    dbms_output.put_line('vlastnosti jsou totozne');
     ELSE
    dbms_output.put_line('vlastnosti nejsou shodne');
     END IF;
commit;
end;
```

Obr. 2. Stránka uživatelské příručky

Na obrázku 3 je ukázka jedné metody z uživatelské příručky. V horní části (světle žlutá barva) je popsáno k čemu metoda slouží a jaké jsou její možné výstupy. Ve spodní části je uveden příklad, který je názornou ukázkou použití dané metody.

Jednotlivé metody jsou spolu s příklady v příručce roztříděny podle toho ke kterému objektovému typu náleží. Jedinou vyjímkou jsou univerzální metody, které se používají pro všechny objektové typy. Tyto metody jsou uvedeny v první kapitole. To je z důvodu, že při popisu některých metod pro konkrétní objektový typ je na tyto metody odkazováno.

V dalších kapitolách jsou nejdříve popsány konstruktory, které se používají pro daný objektový typ a pak teprve jsou popsány metody. Konstruktory jsou vždy 2 a jejich použití je pro všechny objektové typystejné. Proto je názorný příklad jen u objektového typu ORDAudio.

5.2 Tabulky pro vytváření příkladů

Tabulky se v databázových systémech vytváří z důvodu lepšího a přehlednějšího ukládání dat. Aby bylo možné realizovat jednotlivé metody pro práci s databází je nutné nejdříve mít tabulku.

5.2.1 Tvorba tabulky

Pro ukládání dat, ať už ve formě textu nebo souborů, do tabulek je nejdříve potřeba tabulku vytvořit. To se provádí příkazem

```
CREATE TABLE "SYSMAN"."DAV_POKUS"
  (    "ID" NUMBER,
       "DATA" "SYSMAN"."ORDAUDIO"
  )
```

- sysman je uživatelské jméno databáze
- dav_pokus je název tabulky
- id je název sloupce s datovým typem number (číslo)

- data je název sloupce, kde se budou ukládat objekty (v tomto případě audio – je potřeba ORDAUDIO napsat velkými písmeny jinak dojde k chybě (neplatný datový typ))

Počet sloupců je libovolný, záleží jaké informace je potřeba do tabulky uložit. Před vytvořením tabulky je potřeba si pořádně rozmyslet jaké datové typy bude do tabulky potřeba ukládat. Když se mají ukládat například pouze obrázky použije se objektový typ ORDImage. Ale v případě, kdy je potřeba do jednoho sloupce v tabulce ukládat různé typy multimediálních dat je potřeba použít objektový typ ORDDoc, který umožňuje ukládat různé typy dat (obrázky, hudba, PDF apod.).

5.2.2 Vkládání dat do tabulky

Pro vkládání dat do tabulky se použije příkay *INSERT INTO*. V případě datových typů jako INTEGER nebo VARCHAR2 se hodnoty zadávají přímo. V případě objektových typů se pro inicializování musí v tabulce použít konstruktor pro daný objektový typ.

Jsou 2 druhy konstruktorů, init() bez parametrů a init(typZdroje, umístěníZdroje, jménoZdroje) s parametry. Oba konstruktory inicializují instanci daného objektového typu.

BEGIN

INSERT INTO tab_video (id, video)

VALUES (1, ORDSYS.ORDVideo.init());

COMMIT;

END;

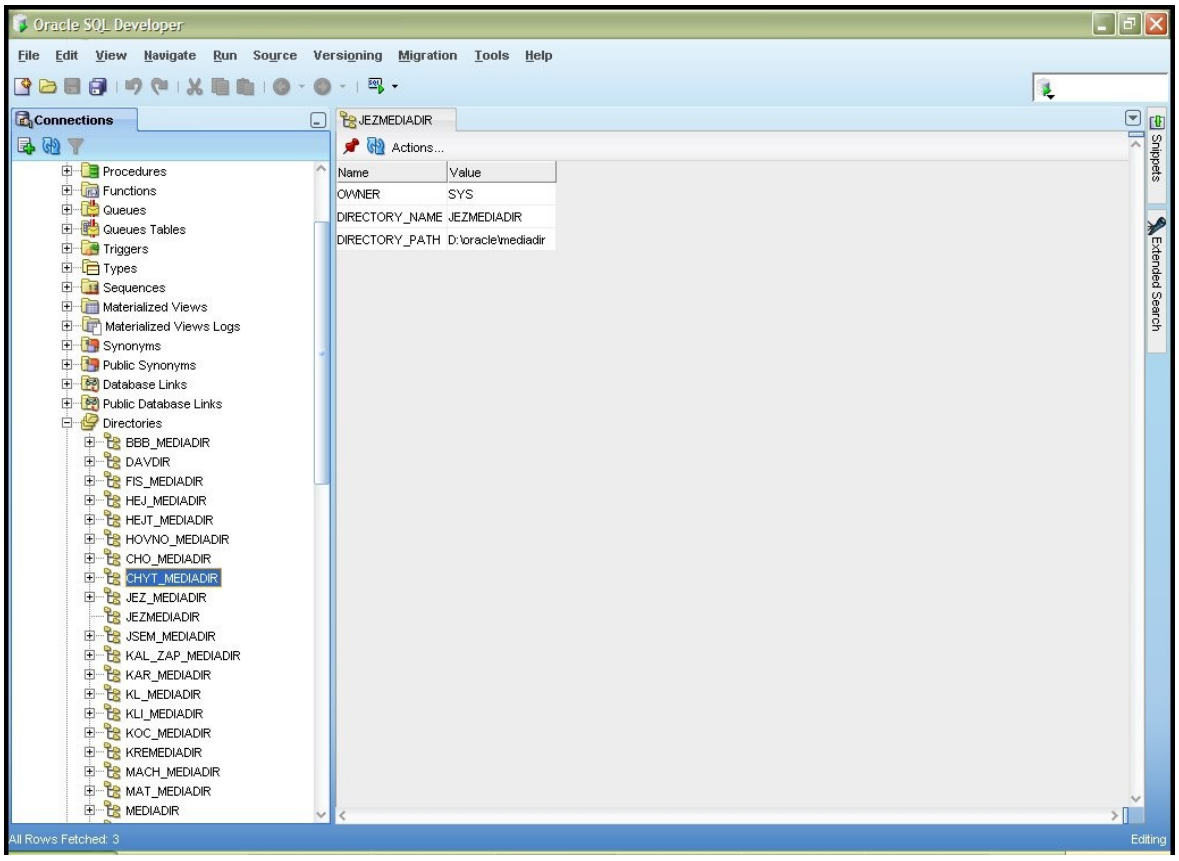
- tab_video je název tabulky do které se ukládá (v závorce jsou názvy sloupců)
- VALUES – hodnoty, které se zadají v jednotlivých sloupcích, musí být oddělené čárkou
- COMMIT – příkaz, který pošle data do databáze ke zpracování, bez tohoto příkazu by se zadané hodnoty do databáze neuložily

Rozdíl mezi konstruktory je v tom, že init() bez parametrů se používá jen pro inicializaci, ale vlastnosti objektu se do atributů neukládají. To lze provést pomocí konstruktoru init() s parametry, který ukládá do atributů ty hodnoty, které jsou zadány v jeho parametrech. Samotné objekty se ukládají pomocí metod import() a importFrom().

Po inicializaci objektu v tabulce, je již možné zadávat pomocí set metod různé hodnoty atributů, ale není to vhodné, protože ještě není vložený žádný objekt a jeho vlastnosti, které by s atributy souhlasily. Proto se hodnoty atributů zadávají až po vložení objektu a je potřeba zadávat správné hodnoty do atributů. Je to z důvodu, že metody get, které zjišťují vlastnosti daného objektu, to zjišťují právě ze zadaných atributů.

Dříve než se použije konstruktor s parametry, kde se zadávají hodnoty typ zdroje (file), umístění zdroje (adresář) a jméno zdroje (název souboru) je nutné si ověřit zda daný adresář existuje. V případě, že daný adresář neexistuje nelze tento konstruktor použít, dokud se potřebný adresář nevytvoří.

5.2.3 Vytvoření a výběr adresáře



Obr. 3. Zobrazení adresářů

Je možné si vybrat již dříve vytvořený adresář. Seznam adresářů lze zjistit rozbalením nabídky Directories. Pro získání informací, stačí na vybraný adresář dvakrát kliknout. Objeví se tabulka, kde je důležitá položka DIRECTORY_NAME. To je název adresáře, který se používá jako parametr konstruktoru init(). Položka DIRECTORY_PATH říká, kde je daný adresář uložen na disku.

Když není vhodný žádný z adresářů v seznamu je možné si vytvořit vlastní adresář. To se provádí pomocí příkazu *create directory NOVYDIR as 'd:\oracle\mediadir'*.

- NOVYDIR – název nového adresáře
- 'd:\oracle\mediadir' – úplnácesta

Po vytvořenínovéhoadresáře je již možné vkládat do databáze nové objekty.

5.2.4 Příkazy pro vložení dat

Když je již v tabulce inicializován vhodný objektový typ, je možné začít vkládat samotná multimediální data. K tomu slouží metody import() a importFrom(). Před použitím metody import() je nutné použít metodu setSource(), kde se nastaví údaje o typu zdroje, jeho umístění a jménu souboru. V případě použití metody importFrom() se dané údaje zadají přímo v parametru této metody.

Při zadavání typu zdroje se zadá název *file* (soubor), umístění zdroje je adresář, který již byl v databázi vytvořený nebo který byl vytvořen pro toto konkrétní ukládání dat. Jméno zdroje je samotný název souboru, který se má do databáze uložit.

Je vhodné dříve než se metody import() a importFrom() použijí, přidat nakonec kódu příkaz ROLLBACK. Tento příkaz (operace) v případě chyby zpracování vrací databázi do předchozího stavu. Vrátí zpět všechny změny až do místa kdy byl zadán poslední příkaz BEGIN. To je důležité pro zachování datové integrity. V případě, že všechno proběhlo v pořádku, příkaz ROLLBACK se vymaže a data se po dalším zpracování SQL dotazu uloží do databáze.

Pro objektové typy ORDAudio, ORDVideo a ORDImage se příkazy import() a importFrom() používají stejně. Rozdíl je u objektového typu ORDDoc, pomocí kterého lze vkládat do databáze všechny druhy dat. U tohoto objektového typu se jako poslední

parametr ještě zadává hodnota typu BOOLEAN. Ta nám určí, zda se má při ukládání dat do databáze zavolat metoda setProperties(). V případě zadání hodnoty FALSE se metoda setProperties() volat nebude, v případě hodnoty TRUE se metoda zavolá. To způsobí, že při ukládání dat se rovnou načtou jejich vlastnosti (formát dat, typ MIME a velikost dat) a uloží se do atributů objektu.

5.3 Příklady pro ukládání hodnot do atributů

Hodnoty do atributů se zadávají z důvodu, aby v případě, kdy je potřeba zjistit nějaké informace o daném objektu se nemuseli získávat přímo z daných multimediálních dat. Výhodou je rychlejší přístup k informacím o vlastnostech daného objektu a v případě špatně zadaných hodnot nedojde ke změně vlastností, například obrázku.

Pro každý objektový typ jsou definovány různé set metody. Rozdíly mezi metodami jsou způsobené tím, že u obrázků získáváme jiné vlastnosti než třeba u videa. Například informace o délce trvání videa je u obrázků nepoužitelná. Používání těchto metod je ovšem velice podobné.

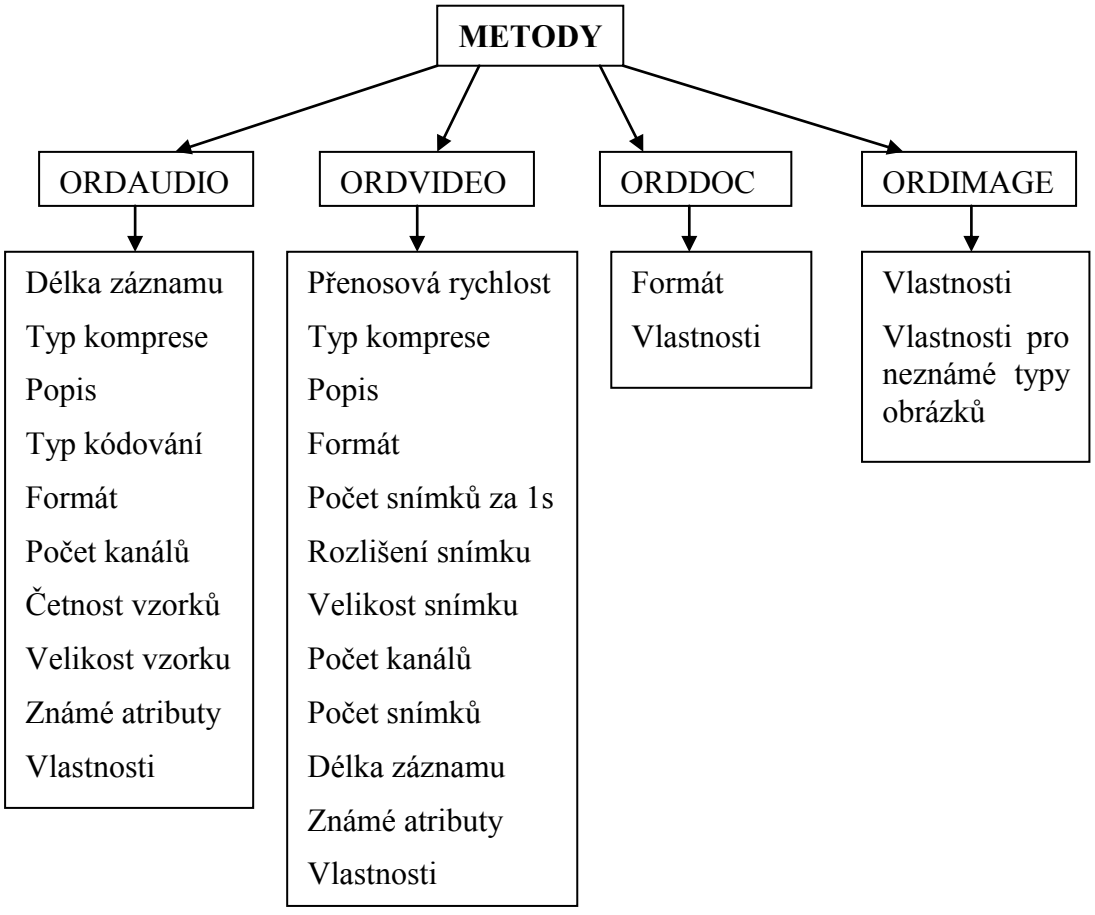
Využívání metod set, kde se nastaví jen požadovaný atribut, je výhodné když dojde ke změně nějakých vlastností daného objektu. Například se může změnit typ formátu z WAVE na MPGA, pak je potřeba změnit danou hodnotu i v atributu objektu. Pro kontrolu v tabulce jestli změna byla provedena úspěšně je nejdříve potřeba tabulku aktualizovat. To se provádí stisknutím tlačítka REFRESH jak je patrné z obrázku 3.

Obr. 4. Tlačítko REFRESH

Na obrázku 4 je graficky znázorněno rozdělení metod SET, které se používají při práci v interMedia. Je vidět, že nejvíce metod je pro práci se zvukovými záznamy a pro práci s video nahrávkami.

Pro všechny objektové typy je jedna metoda společná, ale její použití není u všech objektových typů stejné. Je to metoda setProperties(). Tato metoda si sama zjistí vlastnosti vloženého objektu a vloží je do atributů objektu. To je výhodné, protože při používání ostatních metod je nutné si dané vlastnosti nejdříve zjistit než se budou zadávat do atributů objektu. Navíc při ručním ukládání vlastností může dojít k chybě a tím k nesrovnalostem mezi daty.

Ostatní metody už jsou specifické pro konkrétní objektový typ. Jsou ale i metody, které se používají stejným způsobem, rozdíl je v nastavení jejich parametrů. Například metoda setFormat() je ve své podstatě shodná pro všechny objektové typy, ale pro obrázky se používají jiné typy formátu než třeba pro video. Podobný příklad je i metoda setCompression(), kde se pro audio záznamy používá jiný typ komprese než pro video nahrávky.



Obr. 5. Rozdělení metod SET

5.3.1 Tvorba příkladu pro metodu setFormat()

Pro vytvoření příkladu je použit jazyk PL/SQL. Je to procedurální jazyk od firmy Oracle, který je nadstavbou jazyka SQL. Jazyk má tři části, deklarační, část s příkazy a část pro obsluhu vyjímek. Povinná je pouze část s příkazy, ale v uvedeném příkladu jsou uvedené všechny tři části. Jedná se o metodu setFormat().

```
DECLARE
objekt ORDAudio;
BEGIN
SELECT audio INTO objekt FROM koc_audio_table
WHERE id=2 FOR UPDATE;
objekt.setFormat('WAVE');
UPDATE koc_audio_table  SET audio = objekt
WHERE id=2;
COMMIT;
EXCEPTION
WHEN ORDSYS.ORDAudioExceptions.NULL_INPUT_VALUE THEN
DBMS_OUTPUT.put_line('Vkladana hodnota je NULL');
WHEN OTHERS THEN
DBMS_OUTPUT.put_line('Zachycena vyjímka);
END;
```

Pro uvedený příklad bylo stanoveno, že se má nastavit hodnota atributu formát dat na typ WAVE pro audio objekt, který má hodnotu id 2. Dále je potřeba ošetřit vyjímky. První vyjímka má určit stav, kdy bude zadávaná hodnota NULL. Druhá vyjímka má ošetřit ostatní možné situace.

- nejdříve je v deklarační části deklarována proměnná s názvem objekt a její přiřazen objektový typ ORDAudio
- audio je název sloupce v tabulce a tabulka se jmenuje koc_audio_table
- poté příkazem BEGIN začíná vlastní část s příkazy, kde se nejprve příkazem SELECT vybere sloupec, příkaz FROM určí z jaké tabulky a podmínka WHERE stanoví pro jaký řádek výběr platí
- je potřeba si dát pozor na to, že v takto použitém příkazu SELECT v jazyce PL/SQL je nutné vložit příkaz INTO a do něj umístit objekt, jinak dojde k chybě při kompilaci, výhodou je, že dojde k rychlému získání odpovědi (která obsahuje jeden řádek) na dotaz

- když je takto vybraná položka tabulky, v tomto případě audio objekt, je již možné použít metodu setFormat(), která uloží do proměnné objekt typ formátu WAVE, jako hodnota formátu se dá uložit jakýkoliv text, například „blbost“, ale pak nebude mít nastavení této metody žádný význam a navíc při získávání informací o daném objektu se nezjistí správný typ formátu
- v proměnné je tedy už uložená požadovaná hodnota a je potřeba ji uložit na požadované místo do databáze, to se provede tak, že se provede aktualizace celé tabulky příkazem UPDATE název tabulky
- příkaz SET přiřadí hodnotu výrazu (v tomto případě objektu) do určeného cíle (sloupce), tím dojde k zapsání typu formátu do atributu objektu uloženého v tabulce
- příkaz COMMIT ukončí databázovou transakci a uloží změny pro ostatní uživatele databáze, opačným příkazem je ROLLBACK, který vrací změněné objekty do stavu v jakém byly před započetím transakce – toho se dá využít, když je potřeba otestovat funkčnost nějaké metody, ale přitom neprovádět žádné změny v databázi
- v poslední části příkladu jsou definované tzv. uživatelské vyjímky, ty se nemusí uvádět, ale je to vhodné pro případ, že dojde k nějaké předpokládané chybě, například se v parametru metody místo formátu zadá NULL, tak dojde k vyvolání vyjímky, ze vkládaná hodnota je NULL, v případě, že vyjímka nebude určena dojde k chybě kompilátoru
- vyjímky jsou přesně určené, které lze pro jakou metodu použít
- používají se v podmínce WHEN (když) nějaká vyjímka THEN (potom) proveď výpis na obrazovku
- aby se nemuseli vypisovat všechny možné vyjímky je možné použít v podmínce WHEN označení OTHERS (ostatní) a vypsat na obrazovku, že došlo k vyjímce, tím se ujednodušší práce při definování vyjímek, na druhou stranu se při vyvolání vyjímky nezjistí o jaký typ přesně jde
- na závěr se příkazem END ukončí celý kód
- je důležité ukončit každý příkaz středníkem

Používání ostatních set metod k nastavení jednotlivých atributů je stejné, jen se musí použít vhodná metoda pro uložení daného atributu.

5.4 Příklady pro získání hodnot z atributů

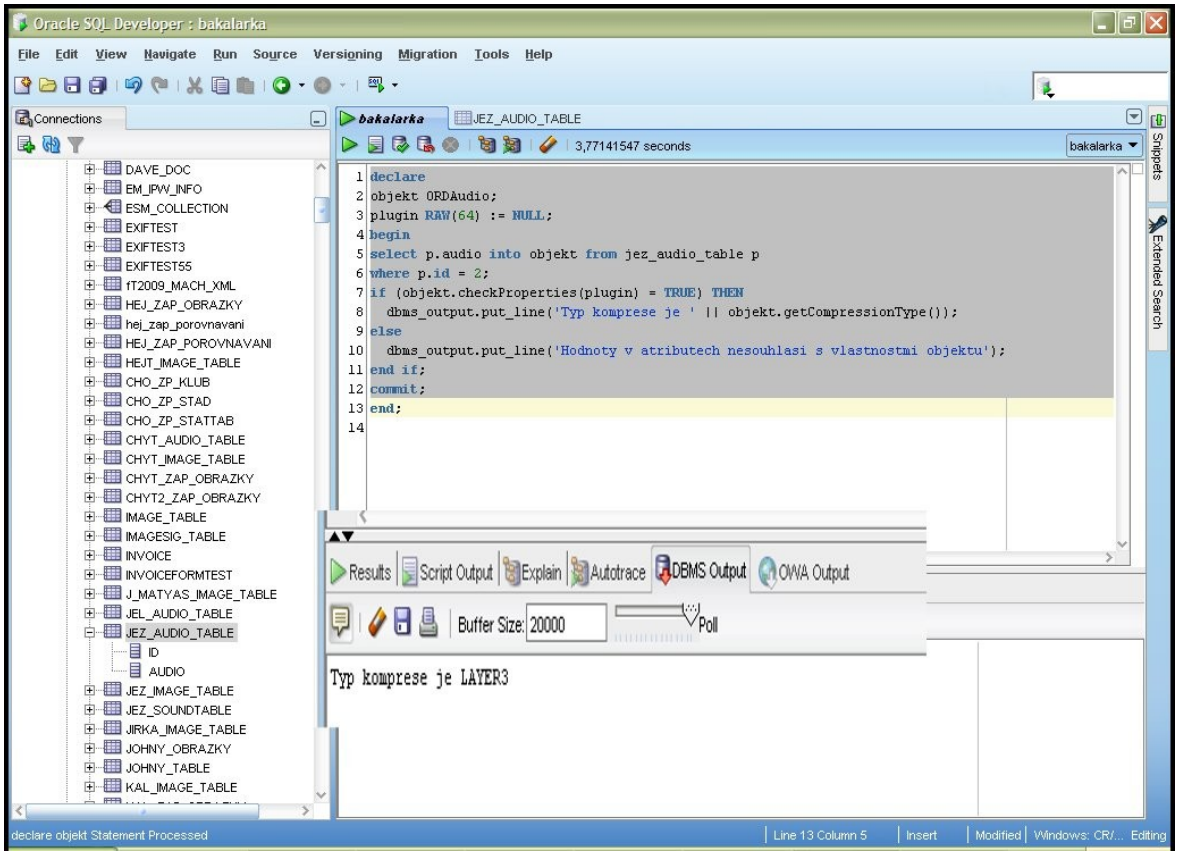
Tato kapitola popisuje jaké jsou možnosti, když je potřeba zjistit nějaké informace o nějakém objektu a jeho vlastnostech. Používají se k tomu metody GET. Stejně jako u metod SET jsou metody GET pro každý objektový typ jiné. Jejich použití je ale podobné. Pomocí nich lze například zjistit jaká hodnota je uložena v atributu typ komprese daného obrázku. Všechny hodnoty, které se zjistí použitím metod GET se získávají z hodnot

atributů. To znamená, že se nejedná přímo o vlastnosti daných dat, ale o hodnoty, které byly uloženy do atributů.

Aby se zabránilo chybám při získávání hodnot jednotlivých atributů je vhodné před zjišťováním hodnot metodou GET použít metodu checkProperties(). Pomocí této metody se zjistí zda zadané hodnoty v atributech odpovídají skutečným vlastnostem dat. Výstupem této metody je pouze hodnota ANO nebo NE. Z toho vyplývá, že v případě negativního výsledku už nelze zjistit, který atribut nesouhlasí s vlastností dat. Prakticky se dá tato metoda využít jako podmínka, která při splnění bude pokračovat ve výpisu hodnot atributů. V případě negativního vyhodnocení se jen vypíše, že dané hodnoty nesouhlasí a dotaz se ukončí aniž by se dále zjišťovala hodnota atributu.

```
if (objekt.checkProperties(plugin) = TRUE) THEN
  dbms_output.put_line(‚Typ komprese je ‚ || objekt.getCompressionType());
else
  dbms_output.put_line(‚Hodnoty v atributech nesouhlasi s vlastnostmi objektu‘);
```

- v podmínce if se otestuje zda dané hodnoty souhlasí
- pokud ano, dojde ke zjištění hodnot atributu typ komprese a vypsaní hodnoty na obrazovku
- pokud ne, na obrazovku se vypíše, že dané hodnoty nesouhlasí



Obr. 6. Okno s výstupem

Ve zvětšené části obrázku je lišta s výsupy. Po provedení dotazu se zobrazí výsledky, které jsou v položce Results (výsledky). Ve zde uvedeném příkladu je k výpisu výsledku zapotřebí otevřít položku DBMS Output. V případě, že se žádný výsledek nezobrazí je ještě zapotřebí nastavit serveroutput na ON.

Obr. 7. Tlačítka

Nastavit serveroutput na ON se provede pomocí tlačítka, které je na obrázku 6 na levé straně. Při jeho opětovném zmáčknutí se serveroutput nastaví zpět na OFF. Když už je serverouput nastavený na ON je zapotřebí znovu provést spuštění dotazu. Teď již dojde k výpisu výsledku na obrazovku. Druhé tlačítko, které je na obrázku 6 slouží k vymazání výsledků v okně.

Tímto je již zpracování metody GET hotovo a výsledek zobrazen. Na stejném principu fungují i ostatní GET metody u všech objektových typů.

U objektových typů ORDVideo a ORDAudio lze zjistit hodnoty všech atributů pomocí jediné metody. Je to metoda getAllAttributes(). Tato metoda vypíše hodnoty všech atributů oddělených čárkou. Výhodou této metody je, že nejprve napíše o jaký atribut se jedná a pak jeho hodnotu. Například *format=WAVE,mimeTYPE=audio/x-wav,….*

Další metodou, kterou lze zjistit hodnotu libovolného atributu je metoda getAttribute(). Na rozdíl od předchozí metody, tato metoda zjistí jen atribut, který se zadá v parametru metody. Stejně jako předchozí metoda se dá použít pouze pro objektové typy ORDAudio a ORDVideo.

Ostatní GET metody již zjišťují hodnoty atributu pro který jsou určeny. Jsou pro každý objektový typ různé. Záleží na tom jaké vlastnosti se pro daný typ dat potřebují a které jsou uloženy.

5.4.1 Tvorba příkladu pro metody getHeight() a getWidth()

Stejně jako u metody setFormat() je použit jazyk PL/SQL. Jedná se o podobný příklad, proto již nebudou uvedeny vyjímky. V tomto případě půjde o to zjistit, jakou šířku a výšku má vybraný obrázek.

```
DECLARE
objekt ORDImage;
vyska INTEGER;
sirka INTEGER;
BEGIN
SELECT p.image INTO objekt FROM image_table p
WHERE id = 2;
-- vyska formatu v pixelech
vyska := objekt.getHeight();
-- sirka formatu v pixelech
sirka := objekt.getWidth();
DBMS_OUTPUT.PUT_LINE('Vyska je ' || vyska);
DBMS_OUTPUT.PUT_LINE('Sirka je ' || sirka);
COMMIT;
END;
```

V této úloze je za úkol zjistit hodnoty atributů o šířce a výšce uloženého obrázku, který má id rovno 2. Získané hodnoty je potřeba uložit do proměnných, pro případ, že by se s nimi mělo dále pracovat. Například pro porovnání hodnot s dalšími obrázky, aby se nemusela pokaždé volat příslušná metoda.

- v deklarační části jsou určenné 3 proměnné
- proměnné objekt je přiřazen objektový typ ORDImage, a proměnným vyska (výška) a sirka (šířka) je přiřazen datový typ INTEGER, to je z důvodu, že výsledne hodnoty proměnných sirka a vyska budou udavat číselný rozměr obrázku v pixelech
- název sloupce je image a název tabulky je image_table
- příkaz SELECT je stejný jako v předchozím příkladu, jen je zde pro znázornění uvedena možnost jak přejmenovat název tabulky na p
- to je výhodné v případě složitějších dotazů, protože názvy sloupců se mohou shodovat, a proto se musí v takových případech před název sloupce ještě připsat název tabulky oddělený tečkou, aby bylo možné sloupec jednoznačně identifikovat
- protože v tomto případě se jedná jen o jednu tabulku není nutné její název umístit před název sloupce, a proto je možné v příkladu zadat název tabulky před název sloupce (image), ale před název sloupce (id) ne
- takto to není vhodné řešit, je dobré používat jeden způsob v celém kódu stejně, protože v případě, že se bude dotaz rozšiřovat může dojít k chybám, které se špatně dohledávají
- když je potřeba napsat komentář, stačí zapsat dvě pomlčky (--) a všechno za nimi do konce řádku už není kompilováno
- při přiřazování hodnoty do proměnné je rozdíl zda hodnotu získáváme pomocí metody nebo přiřazením
- při získávání hodnoty pomocí metody je použit operátor tečka (.) mezi proměnnou a metodou
- v případě přiřazení je potřeba si dát pozor, protože nestačí napsat znak = , ale je potřeba před něj umístit dvojtečku, takže to pak vypadá takto :=
- pro výpis na obrazovku je použit výraz dbms_output.put_line(), kde se do závorky napíše co je potřeba vypsat na obrazovku
- text se musí uzavřít do apostrofu ('například takhle'), ale proměnná se napíše bez nich, pro spojování více částí, například text plus proměnná se musí použít dvojitý znak || , jinak dojde k chybě při překladu
- výraz dbms_output.put_line znamená, že po vypsání obsahu závorky se kurzor přesune na nový řádek, v případě, že je potřeba zůstat na stejném řádku, stačí vynechat slůvko line ve výrazu a ten potom vypadá takto dbms_output.put
- jenom pozor, poslední výraz dbms musí být i s line, jinak se při výpisu nic nezobrazí

Pro získání jiných hodnot atributů (typ formátu, typ komprese) se postupuje stejně. Jen je potřeba při určení datového typu pro proměnnou použít vhodného datového typu. Například pro typ komprese se jedná o datový typ VARCHAR2.

5.5 Práce s obrázky

5.5.1 Úprava obrázku

V databázi Oracle interMedia je také možné měnit přímo vlastnosti objektů vložených do databáze. Jedná se o objektový typ ORDImage, takže lze měnit vlastnosti vložených obrázků. Může se tak změnit typ formátu ve kterém je obrázek uložen nebo třeba velikost komprese, kvůli úspoře paměti v databázi. Na tyto operace se používá metoda process(). Touto metodou se v databázi nevytváří nový objekt, pouze se přepíše ten původní. V případě, že je potřeba zachovat původní objekt, ale zároveň je potřeba z nějakého důvodu vytvořit nový objekt, který už bude uložen s provedenými změnami, je možné použít metodu processCopy(). Obě tyto metody se používají stejně, jenom v metodě processCopy() se jako druhý parametr uvede umístění nově vzniklého objektu. V případě, že se má provést více změn najednou stačí v parametru metody zadat příkazy oddělené čárkou.

Pro kontrolu zda dané změny proběhly v pořádku je možné použít metodu checkProperties(), která zkontroluje zda hodnoty vlastností obrázku a hodnoty atributů souhlasí.

Pro praktické náhledy obrázků, ke kterým se přistupuje přes webové stránky je vhodné vytvořit thumbnail (miniatura obrázku), který se zobrazí ihned při otevření stránky a celý obrázek se zobrazí až při kliknutí na tento náhled. To je praktické z důvodu rychlosti načítaní dané webové stránky. Thumbnail je možné vytvořit přímo v databázi interMedia, a to právě použitím příkazu, který provede potřebné změny. V tomto případě se spíše používá metoda processCopy(), která uloží thumbnail jako nový objekt a zároveň zachová obrázek původní. Pro vytvoření thumbnail se v parametru metody použije operátor *maxScale*, který zachová původní poměr stran obrázku.

objekt1.processCopy('maxScale=32 32', objekt2);

V tomto příkazu je vidět, že zadané hodnoty pro thumbnail jsou 32 pixelů na šířku a 32 pixelů na výšku. Ve skutečnosti ale výsledná hodnota může být nižší. Například když původní obrázek má rozměry 1712 x 2288, tak po zmenšení tímto příkazem má výsledný obrázek (thumbnail) rozměry 23 x 32. Poměr stran (0,7) zůstal zachován, ale velikost obrázku se z původních skoro 4 megapixelů zmenšila na 736 pixelů.

5.5.2 Použití ORDImageSignature a ukázky kódů

Objektový typ ORDImageSignature rozšiřuje možnosti, které lze s obrázky provádět. Pro jeho použití je potřeba v tabulce vytvořit nový sloupec. Stejně jako u ostatních objektových typů je nejdříve potřeba inicializovat instanci. To se provádí pomocí konstruktoru init().

Jedná se o takový podpis obrázku pomocí kterého je možné provádět porovnání mezi obrázky. Nejdříve je ovšem potřeba podpis vytvořit. To se provádí pomocí metody generateSignature().

obrázek_podpis.generateSignature(obrázek);

- obrázek_podpis – proměnná objektového typu ORDImageSignature
- obrázek – proměnná objektového typu ORDImage, jedná se o obrázek, ze kterého se vytvoří podpis

Po vytvoření podpisu je již možné obrázky porovnávat. K tomu jsou určené dvě metody. Jsou podobné, rozdíl je v tom, že metoda isSimilar() pouze říká, zda jsou obrázky podobné. Metoda evaluateScore() říká na kolik procent si podobné jsou. Tohle je možné vyzkoušet tak, že metodou evaluateScore() se zjistí nakolik jsou si podobné a při zadávání hodnoty do parametrů metody isSimilar() se uvede číslo menší a větší. Když se zadá číslo menší, výsledkem bude, že obrázky si nejsou podobné. Když se zadá číslo větší bude vráceno, že obrázky jsou si podobné.

vysledek:=ORDSYS.ORDImageSignature.evaluateScore(obr_podpis1,

obr_podpis2,'color="1.0",texture=0.0,shape=0.0,location=0.0');

vysledek := ORDSYS.ORDImageSignature.isSimilar(obr_podpis1,

*obr_podpis2,'color="0",texture=1,shape=0,location=0',**100**);*

- první ukázka je metoda evaluateScore()
- druhá ukázka je metoda isSimilar()

Rozdíl mezi metodami je, že u metody isSimilar() se uvede jeden parametr nakonec navíc. Ten značí jaký stupeň shody je požadován. Rozsah je od 0 do 100 a čím je číslo větší, tím je větší pravděpodobnost shody. Při zadání čísla 100 bude vždy výsledek, že obrázky jsou shodné.

ZÁVĚR

S potřebou ukládat nebo vyhledávat data se setkáváme v každodenním životě takřka na každém kroku. Princip databází spočívá v tom, aby se tato činnost co nejvíce zjednodušila a zároveň byla přístupná pro co nejvíce lidí. To se velkou mírou používá při práci s internetem, kde je možné vyhledávat konkrétní video nahrávku pomocí popisu (oblíbený server YOUTUBE) nebo třeba v nabídkách jednotlivých obchodů on-line. Přes webové rozhraní se přistupuje právě k databázi daného prodejce a podle toho jak kvalitně je databáze navržena, tak jednoduché nebo intuitivní je využívání jejích možností.

Neustálým vývojem dochází k tomu, že již nestačí získávat informace jen v podobě nějakého popisujícího textu, ale je užitečné doplnit informaci i vhodným obrázkem nebo třeba video ukázkou. Součástí této práce je popis možností, které nabízí databázový systém Oracle interMedia právě při práci s multimediálními daty. Tento databázový systém nabízí další možnosti zjednodušení práce s velkými objemy dat.

Snahou při vytváření této práce bylo připravit pro zájemce o studium tohoto oboru podkladové materiály, kterých mohou využít jako další možnost při snaze o pochopení dané problematiky.

ZÁVĚR V ANGLIČTINĚ

To save and search data has become our daily routine. The objective of databases is to make computing as easy as possible and accessible to everyone. This is largely used with the Internet which enables to search a video after entering its title (popular server YOUTUBE) or with offers of on-line shops. Webservers search the database of the seller and it is the quality of database's design that determines its user-friendliness – its fast and intuitive use of options.

The continuous and rapid development suggests that information in the form of a text is not a sufficient style any more; on the contrary, it is advisable to supplement the text with a corresponding image or a video. A part of this thesis describes ways of working with multimedia which database systém Oracle interMedia provides. This database systém offers various options how work with high volume data can be simplified.

The thesis aims to provide people interested inf this field with some background information that can be used as a source when studying this issue.

SEZNAM POUŽITÉ LITERATURY

[1] Loney, Kevin., Bryla, Bob. Mistrovství v Oracle Database 10g. 1. vyd. Praha: Computer Press, 2006. 704 s. ISBN 80-251-1277-2.

[2] Kyte, Thomas. Oracle: Návrh a tvorba aplikací. 1. vyd. Praha: Computer Press, 2005. 696 s. ISBN 80-251-0569-5.

[3]Procházka, David. Oracle průvodce správou, využitím a programováním nad databázovým systémem. 1. vyd. Praha: Grada Publishing, a.s., 2009. 168 s. ISBN 978-80-247-2762-2.

[4] Introduction to Oracle interMedia [online]. [cit. 2009-12-27]. Dostupný z WWW <https://students.kiv.zcu.cz/doc/oracle/appdev.102/b14302/ch_intr.htm>.

[5] Oracle Database Documentation Library : Oracle interMedia Reference [online].[cit. 2010-03-15].Dostupný z WWW <http://www.oracle.com/pls/db102/portal..portal_db?selected=7>.

[6] Oracle a multimédia? [online]. [cit. 2010-05-02]. Dostupný z WWW <http://www.dbsvet.cz/rservice.php?akce=tisk&cisloclanku=2005072902>.

[7] Referaty:oracle:popis_clientu_pro_praci_s_oracle_databazi_sql_developer [online]. [cit. 2010-04-22]. Dostupný z WWW <http://www.cecak.cz/fel/dba/referaty/oracle/popis_clientu_pro_praci_s_oracle_databazi_sql_developer#pripojeni_k_databazi>.

[8] SQL – jak na dotazy 1. [online]. [cit. 2010-04-22]. Dostupný z WWW <http://interval.cz/clanky/sql-jak-na-dotazy-1/>.

SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK

BLOB	Binary Large Object
PL/SQL	Procedural Language/Structured Query Language
DICOM	Digital Imaging and Communications in Medicine

SEZNAM OBRÁZKŮ