Přístup do ODBC zdrojů dat

Tato kapitola se bude zabývat programováním aplikací využívajících ODBC, a to jak aplikací ve WinBase602 **pracujících s cizími zdroji dat, tak i aplikací v externích jazycích vstupujících do** WinBase602 **přes ODBC.**

Informace o principech ODBC a o využití **WinBase602** jako ODBC serveru jsou součástí manuálu *WinBase602 – Příručka správce*.

Oblast využití **WinBase602** jako zdroje dat (databázový server) pro cizí aplikace je velmi rozsáhlá. V zásadě lze říci, že **WinBase602** plně akceptuje napojení z externích aplikací přes standardizovaná rozhraní ODBC a DAO. Konkrétní zkušenosti hledejte na internetových stránkách ***Software602***.

Programování přístupu k cizím zdrojům dat

Nejprve se soustředíme na nástroje **WinBase602**, které umožňují vyvinout interní aplikaci pracující s cizími zdroji dat přístupnými přes rozhraní ODBC.

Základní mechanismus napojení na zdroj dat a připojení cizí tabulky do aplikace ve **WinBase602** je popsán v manuálu *WinBase602 – Příručka uživatele*.

K práci s cizími zdroji dat a jejich tabulkami potřebujete příslušné ODBC drivery. Tyto drivery nejsou dodávány s **WinBase602**.

Volba funkce zdroje dat

Poté, co vytvoříte spojení na zdroj dat, můžete tomuto zdroji přidělit specifickou funkci v aplikaci. Akcí **Modifikovat** na řídicím panelu lze otevřít dialog pro volbu této funkce:

Ve stavu **Pasivní** lze do aplikace pouze připojovat tabulky zdroje dat a pracovat s nimi normálním způsobem.

Ve stavu **Aktivní** se navíc vytváření nových tabulek nasměruje do tohoto zdroje dat. Návrhář tabulek bude nabízet typy poskytované zdrojem dat a tabulky budou vznikat v tomto zdroji.

Ve stavu **Mapovaný** dosáhnete toho, že se ve zdroji dat vytvoří duplikáty všech tabulek z otevřené aplikace (pokud tam již nejsou) a veškerá práce s tabulkami se přesměruje na ně.

Aktivní nebo mapovaný může být v jedné aplikaci pouze jeden zdroj dat.

Dva modely práce WinBase602 jako ODBC klienta

V tomto a následujícím odstavci detailně popíšeme dva modely práce **WinBase602** jako ODBC klienta. Pro každou aplikaci se na začátku vývoje zvolí jeden z modelů.

První model - přístup k připojeným tabulkám

První model je založen na připojení zdrojů dat a cizích tabulek. Aplikace "sídlí" ve **WinBase602**, ale může pracovat vedle vlastních tabulek také s cizími tabulkami, které jsou do ní připojeny.

Každá tabulka má svůj pevně daný zdroj dat. Proto při jejím návrhu vybíráte z těch typů sloupců, které tento zdroj dat poskytuje. Nepředpokládá se, že by tabulka putovala z jednoho zdroje dat do jiného.

Pokud z prostředí **WinBase602** potřebujete vytvořit cizí tabulku, pak nejprve její zdroj dat označte jako **aktivní** a pak použijete návrhář tabulek. Tabulka se vytvoří nikoli ve **WinBase602**, nýbrž v cizím zdroji dat, a zároveň se do aplikace trvale připojí.

S cizí tabulkou budete pracovat podobně, jako s tabulkami umístěnými ve **WinBase602**. Pokud však budete chtít do ní otevřít kurzor, použijete funkci `Odbc_open_cursor`. Dotazy do cizích tabulek nelze vytvářet návrhářem dotazů.

Pokud aplikaci exportujete a poté importujete na jiném počítači, pak se vytvoří spojení na připojené tabulky. Na tomto počítači musí být předem instalovány potřebné ODBC drivery a zpřístupněny zdroje dat. Předpokládá se, že tabulky v cizím zdroji dat jsou z tohoto počítače dosažitelné.

Druhý model - duplikování tabulek z WinBase602 do cizího zdroje dat

Druhý model vývoje slouží k vytváření aplikací, které při provozu budou mít veškerá data uložena v tabulkách v cizím zdroji dat. Z **WinBase602** se bude při provozu využívat prezentační vrstva: návrhy formulářů a sestav, menu, programy, ale nikoli datové služby serveru.

Druhý model umožňuje vytvářet přenositelné aplikace, které jsou nezávislé na cílovém zdroji dat. Tatáž aplikace pak může být provozována např. s prostými DBF soubory, s **WinBase602** NLM serverem nebo s Ingresovým serverem na UNIXové stanici.

V tomto modelu vytvořte všechny tabulky ve **WinBase602** (žádný zdroj dat není označen jako aktivní) a používáte typů **WinBase602**. Při návrhu tabulek myslete však na to, že

tabulky budou při provozu umístěny jinde, a nepoužívejte multiatributy, sledovací atributy ani jiné rysy, které neposkytuje cílový zdroj dat. Tabulka ve **WinBase602** slouží pouze pro vývoj a testování aplikace jako společný vzor pro tabulky, které mohou vzniknout v různých zdrojích dat.

Také zbytek aplikace vytvoříte normálním způsobem, ale ve formulářích nepoužijete ty rysy, které nejsou k dispozici pro ODBC.

Poté, co je aplikace plně vyvinuta a odladěna v prostředí **WinBase602**, vyzkoušíte její chování na cizím zdroji dat. Za tímto účelem otevřete dialogové okno pro připojování cizích zdrojů dat, označte cílový zdroj jako aktivní a zatrhněte čtverec **Používat tabulky aktivního zdroje dat**. Tím spustíte tyto akce:

- pro každou tabulku z otevřené aplikace se zjistí, zda tato tabulka již existuje v cizím zdroji dat, a pokud ne, pak se vytvoří;
- všechny tyto tabulky z cizího zdroje dat se dočasně připojí;
- dočasně se odpojí tabulky umístěné ve **WinBase602**.

V důsledku toho budou tabulky ve **WinBase602** nahrazeny tabulkami v cizím zdroji dat, a od toho okamžiku se bude pracovat s nimi. Data se přitom mezi tabulkami nepřenášejí. Dotazy, které v tomto pracovním režimu položíte, budou čerpat data z cizích tabulek, kurzory, otevřené normálním způsobem, budou zpřístupňovat obsah cizích tabulek.

Pokud se návrh tabulky ve **WinBase602** změní, je nutno tabulku v cizím zdroji dat zrušit, aby se mohla vytvořit znovu.

Práci s tabulkami v cizím zdroji dat ukončíte tím, že zrušte zatržení čtverce **Používat tabulky aktivního zdroje dat**.

Při importu takto vyvinuté aplikace budete dotázání, na kterém z dostupných zdrojů dat mají vzniknout její tabulky. Návrhy tabulek se pak automaticky přeloží do podmínek cizího zdroje dat a tam budou tabulky existovat. Provozní systém pak bude provozovat aplikaci s takto pevně zvoleným externím zdrojem dat.

Vytváření tabulek v cizích zdrojích dat

Tato sekce popisuje vytváření tabulky v určeném cizím zdroji dat. To je postup, který se často používá v prvním modelu vývoje. Netýká se přenášení návrhu tabulek z **WinBase602** do cizích zdrojů, na němž je založen druhý model. Takto vytvořená tabulka nebude přenositelná do jiných zdrojů dat.

Zdroj dat, v němž chcete vytvořit nebo modifikovat tabulku, musíte označit jako **Aktivní** v dialogovém okně otevíraném akcí **Modifikovat** na řídicím panelu.

Je-li některý zdroj dat aktivní, pak návrhář tabulek pozmění svoje chování takto:

- budou se nabízet ty typy sloupců, které existují ve zdroji dat;

- bude možno zadat specifické údaje, které přísluší ke sloupci v závislosti na jeho typu (např. délku u řetězců);
- nebude možno označit sloupec jako multiatribut;
- vlastnosti indexů budou určeny zdrojem dat (obecně ODBC zná pouze unikátní indexy, ale některé zdroje dat naopak implementují všechny indexy jako neunikátní);
- nebude možno zadávat pravidla referenční integrity;
- v některých zdrojích dat nebude možno zadávat integritní omezení ani implicitní hodnoty.

Pokud to zdroj dat umožňuje, lze zadávat jména sloupců obsahující mezery a jiné speciální znaky, nelze však vytvořit jméno delší než 31 znaků.

Vytvořeným indexům se automaticky přidělí jméno, protože to některé zdroje dat (např. ***MS Access***) vyžadují. Některé jiné zdroje dat však nepovolují jména indexů, proto je nutno je ručně zrušit (například ***dBase*** s driverem verze 1.0). V ***dBase*** a ***FoxPro*** budou vytvořené indexy v každém případě neunikátní.

Vytvoříte-li z **WinBase602** tabulky v cizím zdroji dat, pak tato tabulka bude automaticky připojena do právě otevřené aplikace ve **WinBase602**.

Návrh cizích tabulek z **WinBase602** je omezen obecnými vlastnostmi ODBC API a také specifickými nedostatky různých ODBC driverů. Modifikování definic existujících tabulek některé zdroje dat neumožňují. Pokud tabulky modifikovat lze, pak:

- lze pouze přidávat nebo rušit sloupce, nelze měnit jejich vlastnosti (někdy lze provést jen jednu změnu);
- sloupce se jmény delšími než 31 znaků mají v návrháři místo jména pouze "???";
- nejsou vidět ani nelze měnit implicitní hodnoty sloupců;
- nejsou vidět ani nelze měnit indexy ani integritní omezení;
- nelze popsat způsob přenosu hodnot ze staré tabulky do nové;
- nelze vkládat sloupce na určené místo mezi ostatní sloupce.

Práce s formuláři do cizích dat

Formuláře, vedoucí do cizích zdrojů dat, poskytují všechny funkce, jako normální formuláře směřující přímo do **WinBase602**, s některými výjimkami.

Co lze dělat...

Ve formulářích do cizích zdrojů dat lze:

- vkládat a rušit záznamy a přepisovat hodnoty sloupců v souladu z právy určenými zdrojem dat. Lze editovat texty, pracovat s obrázky a s OLE objekty, pokud zdroj dat tyto typy sloupců obsahuje.
- klást dotazy pomocí QBE a uspořádávat záznamy ve formuláři.
- vytvářet subformuláře a synchronizované formuláře. Relační synchronizace funguje bez omezení. Pokud jsou dva formuláře (např. subformulář se superformulářem) syn-

chronizovány na stejný záznam, pak oba musí mít stejný kurzor. Proto ve vlastnostech subformuláře zapněte **Použít kurzor ze superpohledu** a při otevírání synchronizovaného formuláře z programu použijte příznak `PARENT_CURSOR`.

Co nelze dělat...

Ve formulářích do cizích zdrojů dat nelze:

- pracovat s multiatributy nebo s historii hodnot;
- otevírat relační formuláře.

Další vlastnosti

Některé vlastnosti formuláře závisí na použitém ODBC driveru. Pokud ve formuláři zadáte setřídění záznamů a vložíte nový záznam, pak v závislosti na vlastnostech driveru se tento záznam může ocitnout buď na konci za ostatními záznamy nebo zatříděn na správném místě. Pokud ve formuláři zadáte podmínky pro výběr záznamů a vložíte záznam, který těmto podmínkám nevyhovuje, pak se tento záznam sice vloží, ale v závislosti na vlastnostech driveru může zmizet a objevit se až po odstranění omezujících podmínek.

Zamykání

Pokud ODBC driver podporuje zamykání záznamů, pak i s formulářem do cizích dat lze pracovat v režimu, v němž je vybraný záznam vždy zamčen pro čtení. Nelze však zamknout všechny záznamy. Při otevření textového editoru nebo vázacího formuláře dochází k automatickému zamčení běžného záznamu.

Práce s cizími tabulkami z programovacího jazyka

S cizími tabulkami lze z prostředí **WinBase602** pracovat prostřednictvím *ODBC kurzorů* stejným způsobem, jako při použití běžných kurzorů, až na několik (níže popsaných) výjimek.

Práce s daty se liší podle toho, zda je zapnuto **Použití tabulek z aktivního zdroje dat** či nikoli. Pokud ano, pak funkce **`Open_cursor`**, **`Open_sql_cursor`**, **`Open_sql_parts`** **`Open_cursor_direct`** a **`SQL_execute`** jsou automaticky přesměrovány na aktivní zdroj dat. To znamená, že vytvoříte-li programem tabulku, vznikne v aktivním zdroji dat, a otevřete-li kurzor, bude to automaticky ODBC kurzor, pracující s cizími tabulkami. Mají-li být změny provedené SQL příkazem viditelné na řídicím panelu aplikace, je nutné po volání funkce **`SQL_execute`** volat funkce **`Relist_objects.`**

Poněkud složitější postup musíte zvolit, pokud pracujete s tabulkami **WinBase602** (není zapnuto použití tabulek z aktivního zdroje dat) a chcete vedle toho zasáhnout také do cizích tabulek. Pak musíte otevřít ODBC kurzor níže popsaným způsobem:

ODBC kurzor se deklaruje jako proměnný kurzor a otevírá se funkcí: **`Odbc_open_cursor.`** Číslo spojení na zdroj dat potřebné pro otevření ODBC kurzoru lze získat jednou ze tří funkcí: **`Odbc_find_connection`**, **`Odbc_create_connection`**, **`Odbc_direct_connection`**. Aplikační program by měl volat nejprve funkci **`Odbc_find_connection`** a vrátí-li nulu, pak teprve funkce **`Odbc_create_connection`** nebo **`Odbc_direct_connection`**. Spojení vytvořená

funkcemi **Odbc_create_connection** a **Odbc_direct_connection** zaniknou při skončení běhu programu.
Ve vnitřním programovacím jazyce lze ODBC kurzor použít pro čtení a zápis hodnot sloupců a pro čtení a zápis celých záznamů. Ke sloupcům lze přistupovat pomocí typu záznam uvedeného v deklaraci kurzoru nebo beztypovým způsobem.

ODBC kurzor se dá použít ve funkcích:

- **Read, Write**, **Read_record**, **Write_record**;
- **Delete**, **Insert**, **Append**;
- **Rec_cnt**, **Delete_all_records**;
- **Open_subcursor**, **Restrict_cursor**, **Restore_cursor**, **Close_cursor**;
- **Read_lock_record**, **Read_unlock_record**, **Write_lock_record**, **Write_unlock_record**;
- při otevírání formuláře a tisku sestav (pro přesměrování).

Funkce **Write**, **Delete** a **Insert** mohou nefungovat ve spolupráci s některými staršími ODBC drivery. Drivery, které plně podporují API funkci **SQLSetPos** by neměly činit obtíže. Funkce pro zamykání záznamů pracují s jediným druhem zámků, který nabízí ODBC, proto funkce **Read_lock_record** dělá totéž co **Write_lock_record**.

ODBC kurzor se nedá použít ve funkcích:

- **Undelete**;
- **Read_lock_table**, **Read_unlock_table**, **Write_lock_table**, **Write_unlock_table**;
- **C_sum**, **C_max**, **C_min**, **C_avg**, **C_count**, **Look_up**;
- **Translate**, **Super_recnum**, **Add_record**;
- **Set_fcursor**, **Relate_record**.

ODBC kurzor se zavírá funkcí **Close_cursor**. Při skončení běhu programu ve vnitřním jazyce se zavřou všechny neuzavřené kurzory.

Příklad:

```
var
  c : cursor;
  conn, rec : integer;
begin
  conn:=Odbc_find_connection('WinBase602: D:\WB602');
  if conn=0 then halt;
  if Odbc_open_cursor(conn, c,
      'SELECT * FROM odbc_table') then halt;
  if not Rec_cnt(c,rec) then
     Info_box('Počet záznamů', int2str(rec));
  rec:=Insert(c);
  c[rec].attr:=56;
```

```
  if not Rec_cnt(c,rec) then
     Info_box('Počet záznamů', int2str(rec));
  if Restrict_cursor(c, 'attr > 30') then halt;
  if not Rec_cnt(c,rec) then
     Info_box('Počet záznamů', int2str(rec));
  Restore_cursor(c);
  if not Rec_cnt(c,rec) then
     Info_box('Počet záznamů', int2str(rec));
  Close_cursor(c);
end.
```

Zasílání SQL příkazů cizím zdrojům dat

Zasíláte-li pomocí příkazu **`SQL_execute`** příkaz v jazyce SQL cizímu zdroji dat, pak musíte respektovat jeho specifickou implementaci SQL. V žádném případě nelze využívat rozšíření zavedených **WinBase602**.

Obsahuje-li SQL příkaz řetězec znaků, pak je nutno jej uvést v apostrofech (v souladu se syntaxí SQL), nikoli v uvozovkách (rozšíření **WinBase602**).

Specifické vlastnosti některých ODBC driverů

dBase a FoxPro

- u numerických typů nelze zadat počet číslic před ani za desetinnou tečkou;
- předpokládá se, že obsah souboru je v LATIN 2, a provádí se automatické překódování;
- při zadání data před 1.1.1753 je ohlášena chyba "ODBC File Library: Invalid date";
- indexy vytvořené v tabulkách budou vždy neunikátní;
- při použití driveru verze 1.0 je nutno při vytváření tabulky s indexem zrušit jméno indexu.

Microsoft Access

- některé chyby jsou hlášeny zcela srozumitelně, zatímco jiné pouze jako "Jet error" a číslo;
- driver verze 1.0 pracuje s texty pouze do délky 256 znaků, při čtení nebo zápisu textu větší délky hlásí chybu.

Další omezení práce s cizími tabulkami

V současné době nelze ve **WinBase602** provádět s cizími tabulkami tyto operace:

- import a export návrhu tabulky a dat do/z tabulky;

- přejmenování a duplikování tabulky;
- statistika záznamů v tabulce;
- navrhovat dotaz do cizích tabulek;
- navrhovat SQL operace nad cizími tabulkami.

Modifikovat návrh cizí tabulky lze pouze tehdy, pokud to umožňuje příslušný ODBC driver.

ODBC a síťová práce WinBase602

Mechanismus ODBC je plně zajišťován prezentační vrstvou **WinBase602**. Proto nezávisí na tom, zda používáte server běžící pod ***Windows*** nebo jako NLM. Uvažujme případ, že několik klientů **WinBase602** pracuje se společným databázovým serverem. Popis připojení cizích tabulek je jako součást aplikace uložen na serveru a je sdílen všemi klienty.

Pokud klienti pracují s různými aplikacemi, pak každý si vytváří přístup k cizím tabulkám sám a serveru se tato činnost nijak netýká.

Pokud klienti pracují na stejné aplikaci, pak používají stejné cizí tabulky a stejný popis jejich připojení. Proto aby klienti mohli pracovat:

- zdroje dat všech cizích tabulek musí být přístupny z každého klienta;
- popis přístupu k těmto zdrojům dat a jejich tabulkám musí být pro každého klienta stejný.

Přepokládejme například, že aplikace pracuje z tabulkami uloženými v ***Ingresu***. Pak každý klient musí mít na svém počítači software, který umožňuje přístup k ingresovému serveru: ODBC driver Ingresu a komunikační program.

Pokud aplikace pracuje např. s DBF souborem, pak:

- soubor musí být umístěn na síťově sdíleném disku;
- tento disk musí být na všech klientských počítačích stejně namapován.

ODBC napojení Visual Basicu na WinBase602

V této části si popíšeme, jak lze z programu ***Visual Basic*** přistupovat k datům uloženým v tabulkách **WinBase602** pomocí standardního rozhraní ODBC.

Budou diskutovány dva způsoby práce, odlišující se složitostí:

- Do formuláře VB se vloží komponenta nazývaná Data control a ta se napojí na jednu konkrétní tabulku nebo odpověď na dotaz z **WinBase602**. Další komponenty formuláře (např. DBGrid, TextBox aj.) se provážÍ (angl. *bind*) na výše uvedený Data control

- tím je zaručeno, že po spuštění projektu se v těchto složkách objeví data z **WinBase602**. Pro zpřístupnění dat není potřeba psát žádný kód.

- V programovém modulu (*.BAS) se provede napojení prostředky jazyka. Čtení hodnot i zápis změn pak probíhá pomocí metod příslušejících k jednotlivým objektům datového modelu nazývaného DAO (Data Access Object). Nevyžaduje-li to projekt, není potřeba vytvářet formulář.

Oba dva přístupy lze v praxi samozřejmě podle potřeby zkombinovat do jediného projektu.

Požadavky na Visual Basic

Jednodušší variantu lze zprovoznit i v základní variantě 32-bitového ***Visual Basicu*** nazývané *Standard*, druhý způsob vyžaduje rozšířenou variantu *Professsional*. Testování a vytváření aplikací probíhalo s verzí 4.

Zobrazení dat ve formuláři

Vytvoření formuláře, který zobrazuje data z **WinBase602**, je poměrně jednoduché. Nejdůležitějším krokem je vložení Data controlu do formuláře a jeho nastavení. Nastavit se musí tyto vlastnosti (*properties*):

- do řádku **Connect** zapište řetězec tohoto tvaru (včetně velikosti písmen a interpunkčních znamének):

  ```
  ODBC;DSN=WinBase602: Moje databáze;UID=anonymous; ↵
  PWD=; SCHEMA=Příklady;SERVER_ACCESS=Moje databáze;
  ```

  Za zkratkou **DSN** je uveden přesně ten název ODBC zdroje dat, který je uveden v seznamu dialogu ODBC (System data sources), který ukazuje zaregistrované zdroje dat ve ***Windows***.

  UID a **PWD** odpovídají jménu a heslu uživatele, možno ponechat prázdné (nezapomenout na rovnítko a středník).

  Za slovo **SCHEMA** napište jméno aplikace **WinBase602**, kterou budete používat. Měla by to být tatáž aplikace, která je uvedena ve ***Windows*** v definici **WinBase602** ODBC zdroje dat, ale není podmínkou. Zde uvedená aplikace má přednost.

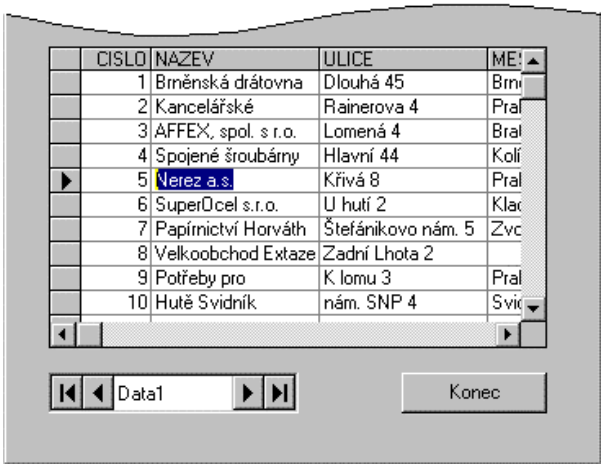
  SERVER_ACCESS udává jméno serveru (databáze), který obsahuje výše uvedenou aplikaci. Musí to být jméno serveru uvedené v definici **WinBase602** ODBC zdroje dat.

- do řádku **RecordSource** zapište jméno tabulky nebo dotazu, jehož obsah chcete zpřístupnit. Můžete také použít nabídku comba - pokud se v combu objeví jména tabulek a dotazů z aplikace z řádku **Connect**, znamená to korektní napojení na zdroj dat. Ne-

objeví-li se, překontrolujte nastavení zdroje dat ve ***Windows*** a odpovídající hodnotu **Connect**.

- V řádku **RecordsetType** zvolte typ napojení: typ **Dynaset** použijte při aktivním editovatelném napojení, typ **Snapshot** pro prohlížecí needitovatelný ale rychlejší režim.

Ostatní vlastnosti volte dle svého uvážení. Formulář s komponentou DBGrid přivázanou na Data control napojený na tabulku **WinBase602** vidíte na obrázku.



Programové napojení

Připojení k tabulce **WinBase602** provedeme např. v rámci jedné funkce. Nejprve je třeba deklarovat proměnné a definovat prostředí (*Workspace*).

```
Function WB_ODBC() As Integer
    Dim WBWorkspace As Workspace
    Dim WBDatabase As Database
    Dim WBRecordset As Recordset
    Dim DataSource As String
    Dim ConnString As String

Set WBWorkspace = Workspaces(0)
DataSource = "WinBase602: Moje databáze"
ConnString = "ODBC;UID=anonymous;PWD=;DATABASE=Příklady"
```

V řetězcích **DataSource** a **ConnString** jsou uvedeny potřebné parametry spojení:

- v DataSource je uvedena hodnota ze seznamu ODBC zdrojů dat registrovaných ve ***Windows*** (totéž, co v předchozím příkladu DSN=...);

- ConnString začíná "ODBC;", je zde také jméno a heslo (UID= a PWD= - může zůstat prázdné), za slovem DATABASE je uvedeno jméno aplikace, s jejímiž tabulkami budeme pracovat. Musí to být tatáž aplikace, která je zvolena u výše uvedeného zdroje dat.

Oba řetězce se použijí jako parametry metody **OpenDatabase**:

```
Set WBDatabase = WBWorkspace.OpenDatabase(DataSource, ↩
False, False, ConnString)
```

Jinou cestou je nechat výběr zdroje dat na VB - použijte následující volání metody **OpenDatabase**:

```
Set WBDatabase = WBWorkspace.OpenDatabase("", False, ↩
False, ODBC;")
```

Od této chvíle je již možné pracovat s objektem **Database** a jeho kolekcemi (*collections*) **TableDefs** a **Recordsets**. Kolekce **TableDefs** obsahuje všechny tabulky a dotazy zvolené aplikace, lze vypsat jména sloupců zdroje dat a jejich typy (kolekce **Fields**), názvy indexů (kolekce **Indexes**) a jejich strukturu (kolekce **Fields**).

Jako příklad uvedeme naplnění listboxu nazvaného ListofTables formuláře TableListForm jmény tabulek a dotazů zvolené aplikace. Použijeme k tomu vlastnosti **Name** objektu **TableDef**:

```
For Each tdf In WBDatabase.TableDefs
      TableListForm.ListofTables.AddItem tdf.Name
Next
```

Pro zpřístupnění obsahu tabulek slouží kolekce **Recordset** a její metody a vlastnosti. Po otevření databáze je tato kolekce prázdná, vložit nový objekt lze metodou **OpenRecordset**.

```
Set WBRecordset = WBDatabase.OpenRecordset(TableName, ↩
dbOpenSnapshot)   ,
```

kde *TableName* může být jméno tabulky nebo dotazu vybrané z předchozího listboxu (nebo lze uvést jméno přímo: "FAKTURY"), konstanta na konci určuje režim otevření: **dbSnapshot** je rychlejší, ale needitovatelný, **dbDynaset** je pomalejší editovatelný.

Na příkladu ukážeme průchod tabulkou s výpisem jména sloupce a jeho velikosti, v případě sloupců proměnné velikosti výpis prvních 20 bajtů.

```
Debug.Print "Průchod tabulkou"
i = 1
Do Until WBRecordset.EOF
  Debug.Print "Záznam č.: "; i
  For Each Col In WBRecordset.Fields
    If Col.Type < 11 Then   'pevná velikost sloupce
      Debug.Print Col.Name, Col.Value
    Else                     'proměnná velikost sloupce
```

```
        Debug.Print Col.Name, "velikost:"; Col.FieldSize
        If Col.FieldSize > 0 Then
          Debug.Print ,"prvních 20 bajtů: "; Col.GetChunk(0,20)
        End If
      End If
    Next
    Debug.Print "---------"
    WBRecordset.MoveNext    ' Posun o jeden záznam
    i = i + 1
  Loop
```

Editace dat probíhá s pomocí metod objektu **Recordset** - **Edit**, **Update**, **AddNew**, **Delete** apod. Změna hodnoty se provede přiřazovacím příkazem do vlastnosti **Value**. Více podrobností naleznete v dodávaných příkladech. V komentářích k příkladům je také uveden kompletní seznam metod a vlastností, které lze pro práci s daty **WinBase602** používat.

ODBC napojení Delphi 2.0 na WinBase602

Ze zkušeností s vývojem aplikací v ***Delphi 2.0*** vyplývají tato doporučení:

Indexy tabulky WinBase602

Bez ohledu na to, má-li tabulka **WinBase602** nějaký index, lze v této tabulce v ***Delphi*** editovat, mazat a vkládat záznamy. Vložíme-li však do tabulky stejné záznamy (resp. shodující se na položkách, na nichž je definováno uspořádání), pak tyto stejné záznamy nelze v ***Delphi*** editovat a mazat.

Metoda Refresh pro Ttable

Chceme-li v ***Delphi*** pro TTable volat metodu **Refresh**, musíme nastavit vlastnost **IdexFieldName,** jinak je vyvolána výjimka: "Tabulka nepodporuje tuto operaci, protože není jednoznačně indexována". Tato výjimka je vyvolána bez ohledu na to, má-li SQL tabulka jednoznačný index či nikoli. Nechceme-li použít k aktualizaci metodu **Refresh**, zavoláme postupně metody **Close** a **Open.** Vlastnost **IdexFieldName** pak nepotřebujeme.

Relist_objects.

Ačkoli volání funkce **Relist_objects** nemá v ODBC viditelný účinek, doporučujeme ji používat po vytvoření nebo zrušení tabulek. Aktualizuje totiž cache objektů.

Aktuální informace vždy hledejte na webovských stránkách ***Software602***. Pro vývoj aplikací v ***Delphi*** poskytuje ***Software602*** poměrně širokou podporu.

ODBC napojení MFC na WinBase602

Ze zkušeností s vývojem aplikací v ***Visual C/C++*** a MFC využívajících rozhraní ODBC vyplývají tato doporučení:

Připojení na server

Vyskytují se problémy s napojením na vzdálený server **WinBase602** pokud není dosud zobrazeno hlavní okno aplikace. Proto vytvářejte spojení na databázový server až po zobrazení tohoto okna.

Odkazy na jména sloupců kurzoru

Generátor tříd (Class Wizard) ve ***Visual C/C++*** vytváří nekorektní přístup k odpovědi na dotaz zahrnující více než jednu databázovou tabulku se stejně pojmenovanými sloupci. Při otevírání dotazu pak server ***WinBase602*** ohlásí chybu syntaxe dotazu.

Podstata problémů spočívá v tom, že když se v tabulkách, na něž odkazuje dotaz, vyskytují stejně pojmenované sloupce, pak generátor tříd zapíše do vytvářeného kódu odkazy na sloupce ve tvaru "`[jméno_tab.jméno_atr]`". Při sestavování dotazu jsou pak hranaté závorky nahrazeny obrácenými apostrofy a vznikne SQL příkaz ve tvaru "**`SELECT` `jméno_tab.jméno_atr` `FROM...`**". Tento zápis je v rozporu s definicí syntaxe SQL v ODBC (přestože je akceptován některými drivery) a způsobí, že **WinBase602** ohlásí chybu syntaxe.

Řešením je odstranit znaky [a] z odkazů na sloupce ze zdrojového programu v jazyce C++ vytvořeného generátorem tříd.