VYSOKÁ ŠKOLA BÁŇSKÁ - TECHNICKÁ UNIVERZITA OSTRAVA

Excel Tables - programování tabulek relačních databází

Doplnění učebních textů pro distanční vyučování

doc. Dr. Vladimír Homola, Ph.D.

Ostrava 2021

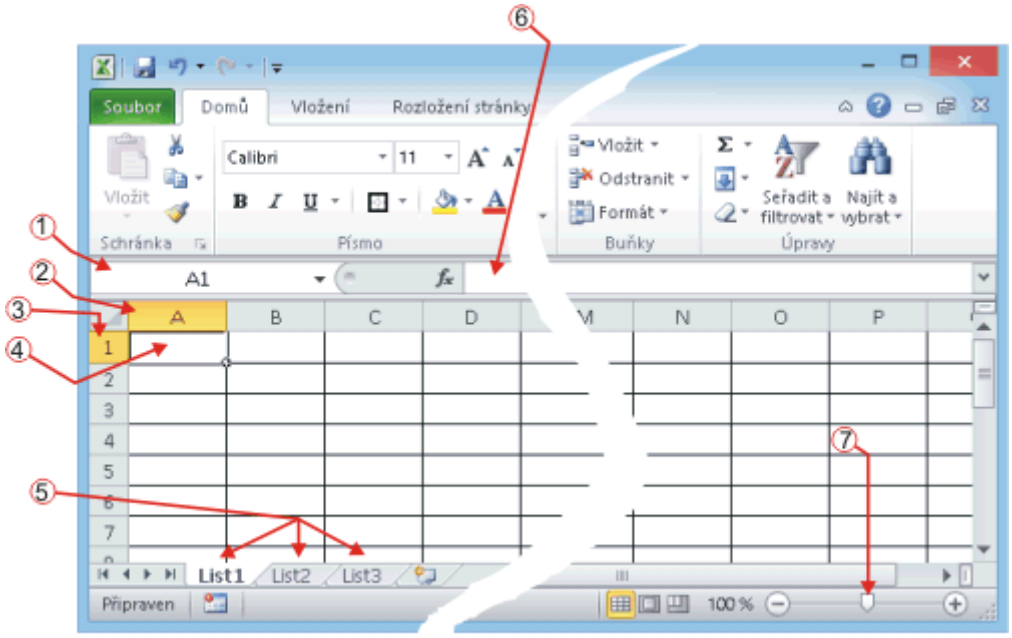
K publikace ISBN 978-80-248-4145-8

OBSAH

Díl V: Programový přístup k tabulkám a relacím

1 Použité konvence, cvičná data

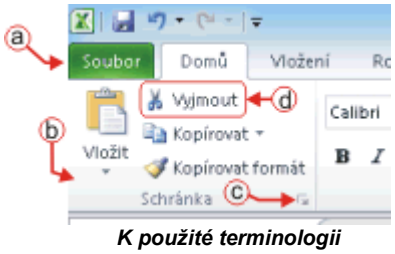
V tomto textu jsou popisovány postupy v prostředí tabulkového procesoru Excel, edice Microsoft Office 2016. Z hlediska uspořádání a ovládání jsou oproti verzi 2010 jen minimální (většinou jen kosmetické) změny, proto pro kompatibilitu se základním textem rozšiřovaných skript (viz [1]) ponecháváme tam uvedené odstavce beze změny. Ve vlastním rozšiřujícím textu už ovšem budou uvedena schémata Excelu 2016.



1. Pole názvů
2. Záhlaví sloupců
3. Záhlaví řádků
4. Aktivní buňka
5. Záložky listů
6. Pole vzorců
7. Lupa

Sešit otevřený v prostředí programu Excel

Pro navigaci při jednotlivých dílčích akcích bude použita následující terminologie:



K použité terminologii

a. Hlavní karty - dříve Hlavní menu, zde aktivní hlavní karta Domů
b. Skupina příkazů - zde skupina příkazů Schránka
c. Tlačítko pro Nástroje skupiny
d. Příkaz - zde příkaz Vyjmout

Tlačítko z bodu c. předchozího seznamu bude v textu zobrazováno symbolem ◢. Pro další ovládací prvky je použita standardní terminologie resp. zobrazení (tlačítko, zaškrtávací pole ☐ resp. ☑, rozvíjecí seznam aj).

Pro vyznačení posloupnosti aktivací návazných ovládacích prvků je použit zápis

`Hlavní karta / Skupina příkazů / Příkaz`

Značí tedy zápis

`Domů / Schránka / Vyjmout`

(viz situaci na předchozím obrázku) požadavek na aktivaci hlavní karty **Domů**, tam vyhledání skupiny **Schránka**, a v ní použít příkaz **Vyjmout**.

Výjimkou je položka **Soubor** hlavního menu, kde na druhém místě je uvedena položka levého panelu a na třetím místě skupina parametrů.

Některé příkazy ve skupině mají pod ikonou nebo vedle textu ještě tlačítko  . Pokud to nevyplyne z kontextu, bude výslovně řečeno, zda se má stisknout ikona nebo toto tlačítko pod (vedle) ní.

V celém následujícím textu se polohovacím zařízením rozumí klasická myš s implicitním nastavením tlačítek jako "levé" a "pravé". Pokud čtenář na svém zařízení používá jiné polohovací mechanismy (dotykovou obrazovky, touch-pad apod.) nebo opačného nastavení tlačítek, nechť pojmy myš, tlačítko, klik a další transformuje vzhledem ke svému zařízení resp. jeho nastavení.

Problematika je vysvětlována na datech cvičného sešitu, který lze stáhnout z adresy

hom50.sweb.cz/CVICGMT.XLSM

2 Úvodní poznámky

Tento doplněk základních skript (viz [1]) vyžaduje znalosti o objektovém programování v jazyku Visual Basic for Applications (VBA) a základní orientaci v objektových třídách knihovny Excel. Všechny níže uváděné příklady jsou zapisovány do modulu nějakého otevřeného projektu (dokumentu Wordu, výkresu Corelu apod.), ale může jím být i známý *ThisWorkbook*. Pokud otevřeným projektem není Excelem otevřený *ThisWorkbook*, musí být referencována (`Tools / References`) objektová knihovna Excelu, např.

```
Microsoft Excel 16.0 Object Library
```

Dále se u čtenáře předpokládá znalost následujících pojmů v rozsahu dle [2]:

- Tabulka relační databáze.
- Tabulka Excelu.
- Relace.
- Kontingenční tabulka.

Pokud v dalším výkladovém textu budou použity identifikátory objektových tříd bez prefixu knihovny, vždy půjde o objektovou třídu z knihovny Excel. Tedy při uvedení identifikátoru např. *Range* půjde o objektovou třídu *Excel.Range*. To neplatí o textu programů, tam se autor snaží používat prefix knihovny Excelu vždy.

Programová řešení uvedená níže by mohla být podstatně kratší, ovšem za cenu totální nepřehlednosti (viz příkazy provádějící totéž, ale generované při záznamu makra). V tomto článku bylo snahou co nejpodrobněji, krok po kroku, ukazovat objektovou hierarchii bez nutnosti složitě pátrat, co vlastně jednotlivé metody a jejich parametry znamenají a instancemi jaké že objektové třídy jsou.

3 Vytvoření a zrušení tabulek

3.1 Tabulky Excelu

Instance objektové třídy *Worksheet* (tj. konkrétní list) obsahuje kolekci *ListObjects* instancí třídy *ListObject*. Právě jedna tato instance je reprezentací jedné Tabulky Excelu dle popisu v [2]. De facto jde o nadstavbu nad obdélníkovou oblastí dat, která disponuje několika často uživatelem vyžadovanými funkcemi (řazení, filtrování apod.). Zrušením této nadstavby se vrací chování dané oblasti dat k "normálu".

Dřívější verze Excelu používaly naprosto konzistentní pojem *Seznam* (což je doslova přeložený anglický pojem *List*), od toho viz objektová třída *ListObject* v objektovém modelu definovaná dodnes. V novějších verzích byl pojem *Seznam* nahrazen pojmem *Tabulka*, a tím se do toho vnesl terminologický zmatek. Proč tomu tak je, to neví snad ani Veliký Bill.

3.2 Vytvoření tabulky Excelu z oblasti dat

Předpokladem je existence obdélníkové oblasti dat splňující definici tabulky relační databáze. Převod této oblasti na tabulku Excelu znamená vložit do kolekce *ListObjects* novou instanci třídy *ListObject*. To zajistí metoda *Add* kolekce. Parametry této metody určují, pro jaký typ oblasti se vlastně nadstavba zavádí (SourceType), kde jsou zdrojová data (Source), a zda první řádek oblasti chápat jako nadpisy sloupců (XlListObjectHasHeaders). Kolekce *ListObjects* je kolekce s klíčem, kterým je námi definované jméno tabulky, to proto musí být v kolekci jedinečné.

Příklad:

```
Sub pgVytvoreniTabulky( _
      qSesit as Excel.Workbook, _
      qJmListu As String, _
      qJmTabulky As String)

   Dim ws As Excel.Worksheet            ' List s připravenými daty
   Dim ls As Excel.ListObjects          ' Kolekce tabulek Excelu listu
   Dim lo As Excel.ListObject           ' Vytvářená tabulka Excelu
   Dim cl As Excel.Range                ' Jedna nebo více buněk oblasti dat
   Dim rg As Excel.Range                ' Celá oblast dat

   Set ws = qSesit.Worksheets(qJmListu) ' Odkaz na list s připravenými daty
   Set ls = ws.ListObjects              ' Kolekce tabulek Excelu na listu

   Set cl = ws.Range("A1")              ' Jedna buňka uvnitř oblasti dat (např. A1)
   Set rg = cl.CurrentRegion            ' Celá oblast dat

   ' Vytvoření nové instance ListObject a vložení do kolekce:

   Set lo = ls.Add( _
      SourceType:=xlSrcRange, _
      Source:=rg, _
      XlListObjectHasHeaders:=xlYes)

   lo.Name = qJmTabulky                 ' Vlastní pojmenování nové tabulky Excelu
   lo.TableStyle = ""                   ' Případné zrušení strakatého formátování

End Sub
```

Parametry podprogramu z příkladu:

- *qSesit*: Nějaký otevřený sešit s daty (např. ThisWorkbook).
- *qJmListu*: Jméno listu v sešitu qSesit, který obsahuje data (např. lVydaje).
- *qJmTabulky*: Vlastní identifikátor, kterým bude pojmenována nová tabulka Excelu (např. tVydaje).

V příkladu se pro zjednodušení předpokládá, že zdrojová data jsou umístěna počínaje adresou A1 (ať těch parametrů není moc).

3.3 Převod tabulky Excelu na oblast dat

Takto nepřesně to nabízí grafické prostředí Excelu. Doopravdy jde o odstranění dané instance *ListObject* z kolekce *ListObjects* a případně i z paměti, a tím i všech vytvořených nadstavbových funkcí. Takto odstranit sama sebe dovede instance třídy *ListObject*, a to svou metodou *Unlist*.

Následující příklad ukazuje odstranění všech tabulek Excelu na konkrétním listu.

```
Sub pgZruseniTabulekListu(qSesit as Excel.Workbook, qJmListu As String)

   Dim ws As Excel.Worksheet            ' List s tabulkami Excelu
   Dim ls As Excel.ListObjects          ' Kolekce tabulek Excelu listu
   Dim lo As Excel.ListObject           ' Rušená tabulka Excelu
   Dim nl As Long                       ' Počet tabulek listu
   Dim il As Long                       ' Parametr cyklu

   Set ws = qSesit.Worksheets(qJmListu) ' Odkaz na list s připravenými daty
   Set ls = ws.ListObjects              ' Kolekce tabulek Excelu na listu

   nl = ls.Count                        ' Počet tabulek listu
   For il = nl To 1 Step -1             ' Cyklus přes všechny prvky kolekce
      Set lo = ls.Item(il)
      lo.TableStyle = ""                ' Odformátování oblasti dat
      lo.Unlist                         ' Odstranění sama sebe
   Next

End Sub
```

Parametry podprogramu z příkladu:

- *qSesit*: Nějaký otevřený sešit s daty (např. ThisWorkbook).
- *qJmListu*: Jméno listu v sešitu qSesit, který obsahuje tabulky (např. lVydaje).

Z logiky programu plyne, že pokud na listu nejsou žádné tabulky, nevadí, cyklus odstranění neproběhne ani jednou.

4 Programové vytvoření připojení

Relace (jednotné číslo) je definována jako logický vztah mezi dvěma datovými zdroji. Pro každý takový zdroj musí být před definováním relace vytvořeno připojení (Connection) aplikace ke zdroji dat. Z Excelu lze vytvořit připojení ke zdroji dat, kterým jsou především data v obdélníkové oblasti splňující definici tabulky relační databáze. Dále musí jít o pojmenovanou oblast. Má-li však toto připojení být využito pro definování relace, musí být navíc taková data rozšířena na tabulku Excelu (viz předchozí kapitola).

4.1 Vytvoření připojení

Instance objektové třídy *Workbook* (tj. konkrétní sešit) obsahuje kolekci *Connections* instancí třídy *WorkbookConnection*. Právě jedna tato instance je reprezentací jednoho připojení na jeden datový zdroj. Vytvořit nové připojení a přidat ho do kolekce umí metoda *Add2* kolekce. Jde o kolekci s klíčem, kterým je námi definované jméno připojení, to proto musí být v kolekci jedinečné.

Níže uvedený příklad je koncipovaný jako funkce odevzdávající nově vytvořené připojení, protože to bude nutno uvést při definování relace.

```
Public Function fcVytvoreniPripojeni( _
      qSesit As Excel.Workbook, _
      qZdroj As String, _
      qName As String _
   ) As Excel.WorkbookConnection

   Const cp As String = "WorksheetConnection_"

   ' Objekty pro připojení
   Dim cs As Excel.Connections            ' Kolekce sešitu všech připojení
   Dim co As Excel.WorkbookConnection     ' Jedno nově vytvářené připojení

   ' Konstruovaná označení:
   Dim ps As String   ' Cesta k sešitu
   Dim js As String   ' Jméno sešitu
   Dim uo As String   ' Úplné označení sešitu
   Dim nm As String   ' Jméno nového připojení

   ' Inicializace
   Set cs = qSesit.Connections

   ' Konstrukce různých označení
   ps = Trim(qSesit.Path)
   If Right(ps, 1) <> "\" Then ps = ps + "\"
   js = Trim(qSesit.Name)
   uo = ps + js
   nm = qName

   Set co = cs.Add2( _
      Name:=nm, _
      Description:="", _
      ConnectionString:="WORKSHEET;" + uo, _
      CommandText:=js + "!" + qZdroj, _
      lCmdType:=7, _
      CreateModelConnection:=True, _
      ImportRelationships:=False)
```

```
    Set fcVytvoreniPripojeni = co

End Function
```

Parametry funkce z příkladu:

- *qSesit:* Nějaký otevřený sešit s daty (např. ThisWorkbook).
- *qZdroj:* Jméno listu v sešitu qSesit, který obsahuje tabulky (např. lVydaje).
- *qName:* Jméno nově přidávaného připojení (např. MojePripojeniNaVydaje).

Příklad je záměrně koncipovaný co nejvíce parametricky. Pokud by byl psán pro konkrétní soubory a další konkrétní podmínky, mohl by být relativně jednoduchý:

```
Public Function fcVytvoreniPripojeni() As Excel.WorkbookConnection

    Dim wb As Excel.Workbook              ' Tento otevřený sešit
    Dim cs As Excel.Connections           ' Kolekce všech připojení sešitu

    Set wb = Excel.ThisWorkbook           ' Pointer např. na tento sešit
    Set cs = wb.Connections               ' Kolekce všech připojení

    Set fcVytvoreniPripojeni = cs.Add2( _
       Name:="MojePripojeniNaVydaje", _
       Description:="", _
       ConnectionString:="WORKSHEET;C:\KURS\indos.xlsm", _
       CommandText:="indos.xlsm!tVydaje", _
       lCmdType:=7, _
       CreateModelConnection:=True, _
       ImportRelationships:=False)

End Function
```

4.2 Odstranění připojení

Odstranit připojení znamená odstranit ho z kolekce *Connections*. Příklad pro odstranění všech připojení v sešitu (parametrem je otevřený sešit):

```
Sub pgZruseniVsechPripojeni(qSesit As Excel.Workbook)

    Dim cs As Excel.Connections
    Dim co As Excel.WorkbookConnection
    Dim nc As Long
    Dim ic As Long

    Set cs = qSesit.Connections
    nc = cs.Count
```

```
    For ic = nc To 1 Step -1
       Set co = cs.Item(ic)
       co.Delete
    Next

End Sub
```

5 Relace

5.1 Vytvoření relace

Instance objektové třídy *Workbook* (tj. konkrétní sešit) obsahuje instanci objektové třídy *Model*, která reprezentuje datový model používaný sešitem. Je přístupná jako vlastnost (Property) se jménem *Model*. Tato instance je inicializována (metoda *Initialize*) při prvním použití datového modelu.

Instance *Model* obsahuje kolekci *ModelRelationships* relací objektové třídy *ModelRelationship*. Vytvořit relaci znamená tedy vytvořit instanci této třídy a přidat ji do kolekce. K tomu slouží metoda *Add* kolekce.

Instance Model dále obsahuje kolekci *ModelTables* tabulek modelu, což jsou instance třídy *ModelTable*. Tato objektová třída jednak disponuje vlastností *SourceWorkbookConnection*, což je pointer na konkrétní připojení (viz výše), a kolekcí *ModelTableColumns*, což je seznam všech sloupců v datovém zdroji.

Předchozí odstavec není pro mechanickou aplikaci níže uvedeného příkladu nutný. Jen ukazuje na postavení tabulek Excelu v datovém modelu a jejich spolupráci s tabulkami modelu. Rovněž vysvětluje, proč při definici relace metodou *Add* postačí jako její parametry zadat jen sloupce: přes ně se dostane k jejich modelovým tabulkám, přes ně k připojením, a přes ně konečně k datům.

```
Public Function fcVytvoreniRelace(     _
      qSesit As Excel.Workbook,              _
      tZdroj As String,   cZdroj As String, _
      tCil   As String,   cCil   As String  _
   ) As Excel.ModelRelationship

   ' Objekty pro relace
   Dim wm As Excel.Model               ' Datový model v sešitu

   Dim rs As Excel.ModelRelationships  ' Kolekce relací
   Dim rl As Excel.ModelRelationship   ' Nově vytvářená relace

   Dim mt As Excel.ModelTables         ' Kolekce tabulek modelu
   Dim tZ As Excel.ModelTable          ' Jedna tabulka modelu - zde jako zdroj
   Dim tC As Excel.ModelTable          ' Druhá tabulka modelu - zde jako cíl

   Dim sZ As Excel.ModelTableColumn    ' Sloupec zdroje, ze kterého vychází vazba
   Dim sC As Excel.ModelTableColumn    ' Sloupec cíle, do kterého směřuje vazba

   Set wm = qSesit.Model               ' Pointer na datový model
   Set rs = wm.ModelRelationships      ' Pointer na kolekci relací
```

```
    Set mt = wm.ModelTables             ' Pointer na kolekci tabulek datového modelu

    Set tZ = mt.Item(tZdroj)            ' Pointer na zdrojovou tabulku
    Set tC = mt.Item(tCil)              ' Pointer na cílovou tabulku

    Set sZ = t1.ModelTableColumns.Item(cZdroj)  ' Sloupec se zdrojem vazby
    Set sC = t2.ModelTableColumns.Item(cCil)    ' Sloupec s cílem vazby

    ' Vytvoření relace, přidání do kolekce a předání jako výsledek funkce:
    Set rl = rs.Add(ForeignKeyColumn:=sZ, PrimaryKeyColumn:=sC)
    Set fcVytvoreniRelace = rl

End Function
```

Parametry funkce z příkladu:

- *qSesit:* Nějaký otevřený sešit s datovým modelem (např. ThisWorkbook).
- *tZdroj:* Jméno tabulky Excelu, která obsahuje zdrojová data (např. tVydaje).
- *cZdroj:* Jméno sloupce ve zdrojové tabulce, ze kterého vychází vazba (např. DRUH).
- *tCil:* Jméno tabulky Excelu, která obsahuje cílová data (např. tDruhy).
- *cCil:* Jméno sloupce v cílové tabulce, do kterého směřuje vazba (např. KOD).

5.2 Odstranění relace

Odstranění relace znamená vypuštění relace z kolekce *ModelRelationships*. Následující příklad ukazuje odstranění všech relací v sešitu:

```
Sub pgZruseniVsechRelaci(qSesit As Excel.Workbook)

    Dim wm As Excel.Model               ' Datový model v sešitu
    Dim rs As Excel.ModelRelationships  ' Kolekce relací
    Dim rl As Excel.ModelRelationship   ' Jedna konkrétní relace

    Dim nr As Long                      ' Počet relací v sešitu
    Dim ir As Long                      ' Parametr cyklu

    Set wm = qSesit.Model               ' Pointer na datový model sešitu
    Set rs = wm.ModelRelationships      ' Pointer na kolekci relací

    nr = rs.Count                       ' Počet relací v kolekci
    For ir = nr To 1 Step -1            ' Cyklus přes všechny prvky kolekce
       Set rl = rs.Item(ir)
       rl.Delete
    Next


End Sub
```

6 Kontingenční tabulka

Níže je uveden příklad funkce, která na základě existujícího datového modelu (viz výše) vytvoří v umístění daném levým horním rohem kontingenční tabulku (třída *PivotTable*). Tu pak předá jako výsledek svého volání. Pro co nejobecnější volání je funkce hojně parametrizována.

```
Public Function fcKontingencniTabulka( _
      qSesit As Excel.Workbook, _
      qJmListu As String, _
      qJmDatModelu As String, _
      qJmKontTab As String, _
      qRadHier As String, _
      qSloHier As String, _
      qMira As String, _
      Optional qPopisekTab As String = "Kontingenční tabulka", _
      Optional qNadpRad As String = "Řádkové kriterium", _
      Optional qNadpSlo As String = "Sloupcové kriterium", _
      Optional qNadpTotal As String = "Celkem", _
      Optional qRadLHR As Long = 1, _
      Optional qSloLHR As Long = 1 _
   ) As Excel.PivotTable

   Dim ws As Excel.Worksheet    ' List, kde se kont. tabulka vytvoří
   Dim cl As Excel.Range        ' Buňka s levým horním rohem tabulky

   Dim ps As Excel.PivotCaches  ' Kolekce mezipamětí
   Dim pc As Excel.PivotCache   ' Mezipaměť sestavy této tabulky
   Dim pt As Excel.PivotTable   ' Reprezentuje sestavu této kont. tabulky v listu
   Dim pf As Excel.PivotField   ' Datové pole kont. tabulky

   Dim cs As Excel.CubeFields   ' Kolekce polí hierarchie nebo míry
   Dim cf As Excel.CubeField    ' Pole hierarchie nebo míry datové krychle

   Set ws = qSesit.Worksheets(qJmListu)   ' Pointer na cílový list
   Set cl = ws.Cells(qRadLHR, qSloLHR)    ' Pointer na levý horní roh

   Set ps = qSesit.PivotCaches  ' Pointer na kolekci mezipamětí kont. tabulek sešitu

   ' Nová mezipaměť pro novou k. tabulku:
   Set pc = ps.Create( _
      SourceType:=xlExternal, _
      SourceData:=qSesit.Connections(qJmDatModelu), _
      Version:=6)
```

```
' Nová tabulka v nově vytvořené mezipaměti:
Set pt = pc.CreatePivotTable( _
   TableDestination:=cl, _
   TableName:=cJmKontTab, _
   DefaultVersion:=6)

Set cs = pt.CubeFields             ' Pointer na kolekci polí hierarchií a měr

Set cf = cs.Item(qRadHier)         ' Pole řádkové hierarchie
With cf
   .Orientation = xlRowField
   .Position = 1
End With

Set cf = cs.Item(qSloHier)         ' Pole sloupcové hierarchie
With cf
   .Orientation = xlColumnField
   .Position = 1
End With

' Pole míry:
Set cf = cs.GetMeasure( _
   AttributeHierarchy:=qMira, _
   Function:=xlSum, _
   Caption:=qPopisekTab)

' Jeho přidání do kolekce polí hierarchií a měr:
Set pf = pt.AddDataField(Field:=cf)

' Některé vlastnosti pole míry:
With pf
   .Caption = qPopisekTab
   .NumberFormat = "# ##0.00"
End With

' Některé vlastnosti vytvořené kontingenční tabulky:
With pt
   .CompactLayoutColumnHeader = qNadpSlo
   .CompactLayoutRowHeader = qNadpRad
   .GrandTotalName = qNadpTotal
End With
```

```
    ' Návrat s předáním vytvořené kontingenční tabulky:
    Set fcKontingencniTabulka = pt

End Function
```

Příklad volání pro datový model popsaný v kapitole Relace:

```
Sub pgVsechnyKontingencniTabulky()

    Dim wb As Excel.Workbook
    Dim pt As Excel.PivotTable

    Set wb = Excel.ThisWorkbook

    Set pt = fcKontingencniTabulka( _
        qSesit:=wb, _
        qJmListu:="lPrazdny", _
        qJmDatModelu:="ThisWorkbookDataModel", _
        qJmKontTab:="ktDruhyObchody", _
        qRadHier:="[tDruhy].[NAZEV]", _
        qSloHier:="[tObchody].[NAZEV]", _
        qMira:="[tVydaje].[KC]", _
        qPopisekTab:="Výdaje celkem", _
        qNadpRad:="Druhy zboží", _
        qNadpSlo:="Obchody", _
        qNadpTotal:="Celkem", _
        qRadLHR:=7, _
        qSloLHR:=3)

End Sub
```

Literatura a další výukové zdroje

[1] Základy použití tabulkových procesorů in: DROZDOVÁ, J., HOMOLA, V.: *Informatika pro Geovědní a montánní turismus, část Informatika.* VŠB - TU Ostrava, 2018. ISBN 978-80-248-4144-1.

[2] HOMOLA, V.: *Excel Tables – analogie tabulek relačních databází* [online]. VŠB - TU Ostrava, 2020, k ISBN 978-80-248-4144-1. [cit. 1. 6. 2021]. Dostupné z http://homel.vsb.cz/~hom50/INFOGMT/INFOXTB.HTM.

[3] Endianness in: TANNENBAUM, A. S., AUSTIN, T.: *Structured Computer Organization.* 6th edition. Boston: Pearson, 2013. ISBN 978-01-329-1652-3.

[4] Institute of Electrical and Electronics Engineers: *IEEE Standard 754-1985 for Binary Floating-Point Arithmetic*, IEEE 1987. Reprinted in SIGPLAN 22, 2, 9-25.

[5] Rounding. In: *Wikipedia: The free encyclopedia* [online]. Wikimedia Foundation, 2003. [cit. 2.9.2020]. Dostupné z: https://en.wikipedia.org/wiki/Rounding.