Denne guide er oprindeligt
udgivet på Eksperten.dk

COMPUTERWORLD

Excel som database i ASP via ADO

Viser hvordan excel kan bruges som database i ASP. Artiklen viser hvordan man henter, indsætter, opdater og sletter fra excel med SQL. Har du en excel fil og gerne vil havde indholdet vist på en hjemmeside men du ikke ønsker at ligge det ind i en database

Skrevet den **03. Feb 2009** af **eagleeye** I kategorien **Programmering / ASP** | ★★★☆☆

Artiklen består af 6 dele:

1. Connecte til Excel med ADO:
Når man connector til en excel fil via ADO gøres det med en connectin streng og man kan enten bruge "Jet OLE DB Provideren" eller "Microsoft Excel ODBC Driver":

```
Jet OLE DB Provideren:

connStr = "Provider=Microsoft.Jet.OLEDB.4.0;Data Source=c:\folder\enfil.xls;Extended
Properties=""Excel 8.0;HDR=Yes;"";"

- eller -

Microsoft Excel ODBC Driver:

connStr = "Driver={Microsoft Excel Driver (*.xls)};FIL=excel 8.0; FirstRowHasNames=1;
DBQ=c:\folder\enfil.xls;"
```

Jeg vil anbefale at bruge den første Jet OLE DB Provideren da den virker mere stabil end ODBC driveren gør. I den connection streng er der specielt en ting som man skal ligge mærke til og det er HDR=Yes. HDR kan sættes til to værdier enten Yes eller No. Forskellen på at sætte den til enten Yes eller No er, ved HDR=Yes bruges teksten i første række som kolonne navne og de kolonne navne kan bruges i forbindelse med et recordset. Ved HDR=No vil første række være data, og der er ikke nogle kolonne navne i recordsetet.

Hvis man bruge Microsoft Excel ODBC Driver kan man også vælge om første række skal bruges som kolonne navne. Det er FirstRowHasNames=1 som måske er mere signede end HDR. Sættes den til 1 bruge første række til kolonne navn, sættes den til 0 hentes første række ud som data. Dog er der en fejl/bug i ODBC driveren så den altid bruger første række som kolonne navne.

I denne artikel vil jeg bruge en excel fil som hedder testfil.xls. Hvis du vil, kan du lave en tilsvarene fil hos

dig selv og prøve koden af på din egen server. Til at starte med har jeg lavet en ny excel fil som hedder testfil.xls. Så har jeg åbnet filen i Excel og lagt dette ind:

```
     A              B
1    Navn       Farvoritis
2    Lise       Gammeldags
3    Ole        Softice
4    Inge       Sodavandsis
```

Det første Ark/Sheet skal have navnet "ark1". A og B er kolonne bogstaverne, og 1 til 4 er række numrene i Excel.

For lige at gøre HDR=Yes eller HDR=No færdig, kan jeg vise et lille eksempel med udgangspunkt i de data som ligger i testfil.xls. Hvis vi antager vi har et stykke som udskriver alt fra excel filen vises resultatet her alt efter hvad HDR er sat til:

Output med HDR=Yes:
Lise Gammeldags
Ole Softice
Inge Sodavandsis

Med HDR=Yes kan man altså angive kolonne navn "Navn" og "Farvoritis" i et recordset: rs("Navn").

Output med HDR=No:
Navn Farvoritis
Lise Gammeldags
Ole Softice
Inge Sodavandsis

Med HDR=No kan man angive kolonnerne med et index, første kolonne har index 0, anden kolonne index 1, osv.. Faktisk laver driveren selv nogle kolonne navne når HDR=No som kan bruges i et recordset, de bliver kaldt "F1", "F2", "F3" osv. Så man kan enten skrive rs(0) eller rs("F1") for at udskrive fra første kolonne.

Jeg har valgt at ligge testfil.xls sammen med .asp filerne så connection til execl filen kommer til at se sådan her ud:

```
<%
filen = "testfil.xls"
Set Conn = Server.CreateObject("ADODB.Connection")
connStr = "Provider=Microsoft.Jet.OLEDB.4.0; Data Source=" & Server.MapPath(filen) & ";"
connStr = connStr & "Extended Properties=""Excel 8.0;HDR=Yes;"";"
Conn.Open connStr
%>
```

2. Hente data fra Excel
Når man vil hente data ud fra excel arket bruges en SELECT SQL sætning. I SQL sætningen skal man normalt angive et tabel navn hvorfra man vil hente sine data. Da excel arket ikke indeholder tabeller er der flere måder at angive hvorfra data hentes.

Angiv et range eller område i excel, denne metode er nok den som kommer tættest på en tabel i andre databaser. For at oprette et område/range åbner man excel filen i excel. Marker med musen det

område/range som skal være tabellen, i dette eksempel markere man A1->B4 så tykke på man boksen som hedder "Boksen Navn" på engelsk "Name Box" hvis man holder musen over. I boksen står der A1. Der skiver man det navn man ønsker ens område skal havde. Det kan være "test", så har området A1->B4 navet test. Det er ikke noget problem at havde flere områder/ranges i samme excel fil det svare til at havde flere tabeller i en Access fil.

Det også muligt at oprette, rette eller slette et range/område fra menuen:
1. klik på menuen Indsæt
2. vælg Navn ->
3. klik på Definer...

Når man har lavet et område/range, så kan man hente data ud med en SQL sætning som man kender fra en database:
SQL = "SELECT * FROM test"

I stedet for at oprette et område/range i excel kan man angive området direkte i SQL sætningen i stedet for tabel navnet. Syntaxen for et område/range angivet i SQL sætningen efter FROM er:

[navnet på arket$]Start kolonne[Start Celle]:Slut kolonne[Slut Celle]

Hvis [Start Celle] anvendes skal [Slut Celle] også anvendes, og omvendt. Bemærk når man skriver et område/range direkte i SQL sætningen skal der klammer [] eller baglæns apostrof ` omkring område/range angivelsen.

Denne henter alt i området A2->B3:
SQL = "SELECT * FROM [ark1$A2:B3]"

Denne henter alt i kolonnerne A->B, dog kun de rækker som indeholder data. Tomme rækker efter sidste post med data medtages ikke:
SQL = "SELECT * FROM [ark1$A:B]"

Denne henter kun celle A2:
SQL = "SELECT * FROM [ark1$A2:A2]"

Man kan udelade navet på arket, og så vil excel automatisk tage det første ark, Denne henter alt i kolonnerne A->B:
SQL = "SELECT * FROM [A:B]"

Desværre kan man kun referere til excel kolonne bogstav i FROM delen. Det ville være dejligt hvis man kunne skrive sådan her, men det virker ikke:
SQL = "SELECT * FROM [A:B] WHERE [B]='bla' ORDER BY [A]"

Så vil man bruge WHERE, ORDER BY eller GROUP BY skal man havde HDR=Yes i connection strengen. Da man godt kan bruge de kolonne navne man selv har angivet i første række, så denne virker:
SQL = "SELECT * FROM [A:B] WHERE farvoritis='bla' ORDER BY navn"

Der er eksempelvis ikke noget problem at lave JOIN af områder ligesom man JOIN'er tabeller i en database. Ved JOIN er man nød til at bruge områder/ranges defineret i excel arket:
SQL = "SELECT * FROM kunder LEFT JOIN ordre ON kunder.id = ordre.kundeid"

Jeg vil helt klart anbefale man bruger HDR=Yes i connection strengen og derved bruger første række til kolonne navne, grunden til det er hvis man vil bruge lidt mere end kun SELECT FROM i SQL sætningen kræver det kolonne navnene indgår i SQL sætningen.

3. Indsætte data i Excel
Det er også muligt at indsætte nye poster i et område/range. Det forgår via SQL med en INSERT INTO som man kender fra andre databaser, når man bruger INSERT INTO skal man bruge kolonne navne og derfor er det nødvendig at havde HDR=Yes i connectoing strengen:
SQL = "INSERT INTO test (navn, farvoritis) VALUES ('Kim', 'Filur')"

Bemærk at Excel selv udvider det område/range som er defineret når man indsætter en ny post og bruger område/range navnet i SQL sætningen. Så når man henter data ud igen og skriver område/range navnet så kommer de nye data også med.

Det er også muligt at angive området direkte i SQL sætningen:
SQL = "INSERT INTO [A:B] (navn, farvoritis) VALUES ('Kim', 'Filur')"

Bemærk dette vil IKKE udvide det område som man har defineret i excel, selvom det bliver tilføjet efter de andre poster som er i området.

Det er også muligt at bruge et recordset .AddNew metoden til at indsætte data:
```
<%
SQL = "SELECT * FROM test"
rs.Open SQL, Conn, 1, 2
rs.AddNew
rs("navn") = "Kim"
rs("farvoritis") = "Filur"
rs.Update
rs.Close
%>
```

4. Opdater data i Excel
Det er også muligt at opdatere data i et område/range. Det forgår via SQL med en UPDATE som man kender fra andre databaser. Når man bruger UPDATE skal man bruge kolonne navne og derfor er det nødvendig at havde HDR=Yes i connection strengen:
SQL = "UPDATE test SET farvoritis = 'Vaffel' WHERE navn = 'Ole'"

Det er også muligt at angive et område/range direkte i SQL sætningen:
SQL = "UPDATE [A:B] SET farvoritis = 'Vaffel' WHERE navn = 'Ole'"

Det også muligt at begrænse områder til nogle få celler den opdater i området A1->B3:
SQL = "UPDATE [ark1$A1:B3] SET farvoritis = 'Vaffel' WHERE navn = 'Ole'"

En anden måde at opdatere på som både virker med HDR=No og HDR=Yes. Hvis man har en excel fil ,og gerne vil opdatere eksempelvis 3 felter i sit område/range og den hver gang skal overskrive de gamle data kan man skrive det sådan her:
```
<%
SQL = "SELECT * FROM rangeName"
rs.Open SQL, Conn, 1, 2
rs(0) = 1
rs(1) = "Noget tekst"
rs(2) = "Noget andet tekst her"
rs.Update
rs.Close
%>
```

5. Slette data i Excel
Det lyder måske let at slette fra Excel og ens første tanke er at lave en SQL sætning som var det en anden database.

SQL = "DELETE FROM test WHERE navn = 'Ole'"

Her kommer Excel i knæ og man finder ud af at Excel ikke er en databasen men et regneark. Det giver denne fejl:
"Deleting data in a linked table is not supported by this ISAM."

Jeg har søgt på nettet og fundet flere artikler i stil med denne. Fælles for dem jeg fandt var de ikke berørte emnet at slette fra excel. Så selvom DELETE via SQL ikke virker, har jeg valgt at beskrive emnet i denne artikel.

Microsoft har skrevet denne side How To Use ADO with Excel Data from Visual Basic or VBA og går man ind og kigger efter DELETE finde man frem til dette:

1. Delete an entire record at once or you receive the following error message:
- Deleting data in a linked table is not supported by this ISAM.

You can only delete a record by blanking out the contents of each individual field.

2. Delete the value in a cell containing an Excel formula or you receive the following error message:
- Operation is not allowed in this context.

3. You cannot delete the empty spreadsheet row(s) in which the deleted data was located, and your recordset will continue to display empty records corresponding to these empty rows.

Så man kan altså ikke bruge en DELETE SQL sætning til at slette en pos, og i punkt 1 skriver MS at man vil få den jeg fik hvis man forsøger.

Dog bliver man lidt positiv stemt da der står at man kan slette en post ved at slette hver enkelt felt. Det strider lidt i mod punkt 3 hvor der står man ikke kan slette en post ved at nulstille eller slette hver enkelt felt, men de poster vil blive som tomme poster.

Det skal komme an på en prøve, og koden bliver hurtigt til en UPDATE SQL som sætte hver kolonne til Null:
SQL = "UPDATE test SET navn = Null, farvoritis = Null WHERE navn = 'Ole'"

Eller man kan buge et recordset til at sætte hver kolonne tom:
```
<%
SQL = "SELECT * FORM test WHERE navn = 'Ole'"
rs.Open SQL, Conn, 1, 2
rs("Navn") = Null
rs("farvoritis") = Null
rs.Update
%>
```

Et hurtigt kig i excel filen efter koden var kørt, viste at rækken stadig var i området, bare med tomme felter. Så det er desværre punkt 3 fra MS side som er rigtigt. De tomme rækker vil blive i arket . Så det er kun indholdet af cellerne som blev slettet, selve posten er der stadig. Jeg har også prøve at sætte kolonnerne til en tom streng "" for at se om det gav en forskel, men det gjorde det ikke (husk Null og "" er ikke det samme).

Så når man ikke får slettet posten vil den også komme med ud når man henter alle poster ud med:
SQL = "SELECT * FROM test"

Hvis man udskriver alle posterne vil det se ud som om der er huller i listen hvis der er nogle poster/rækker som er "slettet" og alle cellerne er tomme. Det kan dog løses ved at tilføje en WHERE del til SQL sætningen så den kun henter de poster ud hvor kolonnerne ikke er tomme. Det er måske ikke en flot løsning, men det er det bedste man kan gøre når man ikke kan slette selve rækken i excel filen med SQL:
SQL = "SELECT * FROM test WHERE (navn is not NULL AND farvoritis is not NULL)"

6. Konklusion
Excel er et regneark og det er ikke det samme som en database. Excel har også nogle begrænsninger når det kommer til SQL som nævnt før i artiklen kan man ikke slette en række via SQL. Jeg vil sige excel kan bruges som "data source" men ikke som database. Så jeg vil kun anbefale at bruge excel hvis man har en excel fil i forvejen og gerne vil trække lidt data ud og vise på en hjemmeside. Det kan jo være man ikke har mulighed for eller ønsker at skifte til en database.

Står man overfor en ny opgave som kræver en database, og selvom man kender bedst til excel, vil jeg klart anbefale at man bruger en database, hvilken databasen det er betyder ikke så meget. Dog har Access den fordel at den er fil baseret lige som excel, så man let kan tage data filen med rundt omkring.

Koden:
Her er koden jeg har brugt til test da jeg skrev denne artikel, du kan copy/paste koden hvis du selv vil prøve at rode lidt med det. Koden virker måske lidt rodet, men den indeholder også 3 forskellige connection strenge, og et stort udvalgt af de SQL sætninger som er vist i artiklen, samt kode til at udskrive data til browseren. Du kan prøve at skifte SQL sætning eller connection streng og se hvad der sker:

```
<html>
<head>
<title>Excel som database i ASP med ADO</title>
</head>
<body>
<%
filen = "testfil.xls"
Set Conn = Server.CreateObject("ADODB.Connection")
Set rs = Server.CreateObject("ADODB.Recordset")

' Jet OLE DB provider HDR = Yes
connStr = "Provider=Microsoft.Jet.OLEDB.4.0;Data Source=" &
Server.MapPath(filen) & ";"
connStr = connStr & "Extended Properties=""Excel 8.0;HDR=Yes;"";"

' Jet OLE DB provider HDR = No
connStrHDRno = "Provider=Microsoft.Jet.OLEDB.4.0;Data Source=" &
Server.MapPath(filen) & ";"
connStrHDRno = connStrHDRno & "Extended Properties=""Excel 8.0;HDR=No;"";"

' ODBC driver
connStrODBC = "Driver={Microsoft Excel Driver (*.xls)};FIL=excel 8.0;
FirstRowHasNames=1;"
connStrODBC = connStrODBC & "DBQ=" & Server.MapPath(filen) &";"
```

```
'Åbner connection vælg en af de 3 linjer
Conn.Open connStr
'Conn.Open connStrHDRno
'Conn.Open connStrODBC

'-------------------------------------------------------------------------
---
'Henter alt i området som hedder test
SQL = "SELECT * FROM test"

'Denne henter alt i kolonnerne A->B:
'SQL = "SELECT * FROM [ark1$A:B]"
'SQL = "SELECT * FROM [A:B]"

'Denne henter alt i området A2->B3:
'SQL = "SELECT * FROM [ark1$A2:B3]"

'Denne henter kun celle A2:
'SQL = "SELECT * FROM [ark1$A2:A2]"

'Indsæt en ny post
'SQL = "INSERT INTO test (navn, farvoritis) VALUES ('Kim', 'Filur')"
'SQL = "INSERT INTO [A:B] (navn, farvoritis) VALUES ('Kim', 'Filur')"

'Update en post
'SQL = "UPDATE test SET farvoritis = 'Vaffel' WHERE navn = 'Ole'"
'SQL = "UPDATE [A:B] SET farvoritis = 'Vaffel' WHERE navn = 'Ole'"

'"Slette" en post ved at indsætte Null eller "" i kolonnerne
'SQL = "UPDATE test SET navn = Null, farvoritis = Null WHERE navn = 'Ole'"


'Åbner recordsetet, her er to muligheder
Set rs = Conn.Execute(SQL)

if rs.State > 0 then
  Response.Write "<table>"
  'Udskriv kolonnenavne
  if not rs.EOF then
    Response.Write "<tr>"
    for each f in rs.Fields
      Response.Write "<td><b >" & f.name & "</b ></td>"
    next
    Response.Write "</tr>"
  end if
  'Udskriv indhold af kolonnerne
  do while not rs.EOF
    Response.Write "<tr>"
    for each f in rs.Fields
      Response.Write "<td>" & f & "</td>"
    next
    Response.Write "</tr>"
    rs.MoveNext
  loop
  Response.Write "</table>"
```

```
else
  Response.Write "Du udførte en INSERT eller UPDATE SQL sætning."
  Response.Write " Prøv at lave en SELECT SQL sætning og reload siden og se
hvad der skete."
end if

if rs.State > 0 then RS.Close
Set RS = Nothing
Conn.Close
Set Conn = Nothing
%>
</body>
</html>
```

Links:
How To Use ADO with Excel Data from Visual Basic or VBA

How To Query and Update Excel Data Using ADO From ASP

Kommentar af jw d. 07. Mar 2007 | 1

Helt fantastisk - lige hvad jeg stod og skulle bruge.

Kommentar af fastwrite d. 05. Nov 2004 | 2

Flot artikel.
Eagleeye har endnu engang vist sig som en dygtig mand.

Kommentar af ellebaek d. 23. Aug 2004 | 3

Super super super :-)

Eagle.: kom endelig med nogle flere artikler der er så gode som denne .-)

Kommentar af bak d. 30. Aug 2004 | 4

Rigtig god artikel. Du har sat dig rigtig godt ind i stoffet. Du burde faktisk også lægge den under excel-kategorien

Kommentar af montago d. 01. Sep 2004 | 5

ret god artikkel !

Kommentar af aspcoder d. 22. Aug 2007 | 6

Hurtig og nem at gå til..