

## MODUL G



<b>Modul G</b> .....	<b>2</b>
FORMEEDITOREN.....	2
1. <i>Værktøjslinien i formeeditoren</i> .....	2
2. <i>Syntaksforklaringerne i formeeditoren</i> .....	3
3. <i>Ofte anvendte funktioner</i> .....	4
4. <i>Opbygning af komplekse formler</i> .....	6
5. <i>Introduktion til @regexpr</i> .....	8
6. <i>Udvælgelse af @funktioner</i> .....	8
<b>Appendiks G</b> .....	<b>10</b>
TRINVIS FORKLARING TIL FØLGENDE FORMEL:.....	10

# Modul G

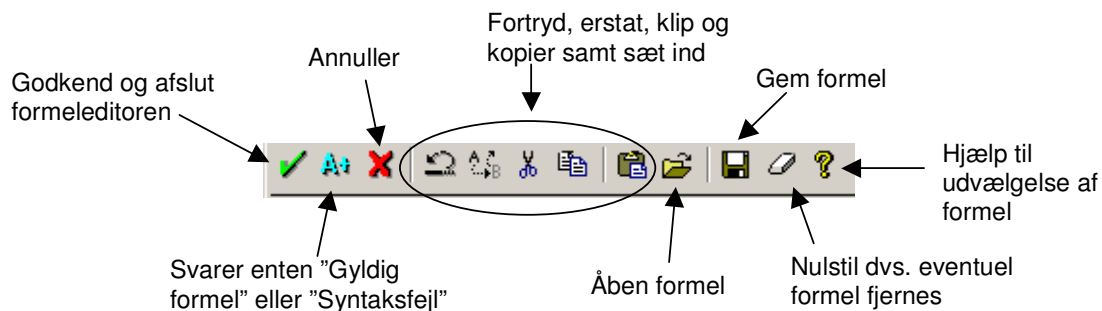
## Formeleditoren

Formålet med dette modul er at give brugeren en dybere viden om formler, uden at gennemgå IDEAs nuværende 84 @funktioner i detaljer. Strategien i modulet er primært at udruste brugeren med metoder, således at denne kan opbygge formler på egen hånd.

Beskrivelsen i model C<sup>1</sup> om formeleditoren brugers flade antages for kendt.

### 1. Værktøjslinien i formeleditoren

Knapperne på værktøjslinien i formeleditoren er beskrevet nedenfor:



**Anbefaling 1:** Benyt gem og åbne knapper til at opbygger din egen, en gruppes eller organisationens formelsamling.

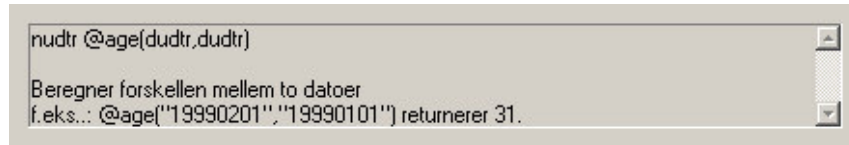
Udformning af en formelsamling kunne f.eks. tage sig således ud:

@funktion	Eksempel	Forklaring
@ctod(en dato som karakterfelt ;"en maske")	@ctod(DATO ;"DD/MM/ÅÅ") <b>Gemt som: C:/form/dato_kon.eqx</b>	Feltet DATO var defineret som et karakterfelt. Cellen "31/12/00" blev konverteret til et datofelt 2000/31/12.
@delete(karakterfelt ; tegnummer; antal tegn)	@insert(@delete(TYPE ; 3;2) ; 3 ; "AA")	Vha. funktionen @delete fjernes 2 tegn startende i position 3 fra karakterfeltet TYPE. Vha. funktionen @insert indsættes AA i 3. position.
@insert(karakterfelt; tegnummer ; "tegn der skal indsættes")	<b>Gemt som: C:/form/slet_ind.eqx</b>	
@val(et karakterfelt)	@val(@remove(@remove(SALDO;"k");"r"))	Feltet SALDO er et karakterfelt der indholder "kr". Funktionen @remove fjerner henholdsvis "k" og "r" og formlen @val konverterer resten til et numerisk felt.
@remove(et felt;"et tegn")	<b>Gemt som: C:/form/tal_fjern.eqx</b>	

<sup>1</sup> Afsnit 2, "Kort om formeleditoren".

## 2. Syntaksforklaringerne i formeledatoren

Syntaksen til en @funktion kan læses nederst i formeledatoren efter at funktionen er markeret. I nedenstående eksempel er der tale om funktionen @age.



I det pågældende eksempel med @age beregner IDEA antallet af dage mellem to datoer i datofelter. Formlen er for eksempel @age(DATO1;DATO2), hvor DATO1=19990201 og DATO2=19990101 for en given post.

I syntaksforklaringerne benyttes forkortelserne: nudtr, dudtr og sudtr. Betydningen af disse ses i nedenstående tabel.

Forkortelse	Betydning
Nudtr	Numerisk udtryk
Dudtr	Datomæssig udtryk
Sudtr	Karaktermæssig udtryk hvor s står for streng

Anførelstegnene i syntaksforklaringen kan godt give indtryk at den korrekte syntaks i ovenstående tilfælde er @age("DATO1",DATO2"), hvilket er forkert.

Anførelstegnene skal f.eks. anføres i formelen @ctod(DATO;"DD/MM/ÅÅ"), hvor "DD/MM/ÅÅ" udgør en maske. Et andet eksempel er @remove(SALDO;"k"), hvor instruktionen "k" i anførelstegn angiver det som skal fjernes.

**Huskeregul:** Instruktioner og masker skal omgives med anførelstegn. Feltnavne skal ikke.



### Handling

- Åben "Detailed Sales File" ved at dobbeltklikke på den.
- Klik derefter på ikonet for **Feltmanipulation**.
- Tilføj et nyt felt med feltnavnet **Tilkarakter**, med længden 5
- Benyt formelen @chr(QTY).
- Godkend formlen.
- Tilføj et yderligere felt med feltnavnet **Tilbage**, som numerisk.
- Benyt formelen @acii(Tilkarakter).
- Godkend formlen.
- Tilføj det sidste felt med feltnavnet kontrol.
- Benyt formelen @if(Tilbage=QTY;"Ja";"Nej").
- Godkend formlen.
- Klik OK i Feltmanipulation.
- Vurder resultatet.

Bemærk at @acii(tom/blank) returneres som 32.

### 3. Ofte anvendte funktioner

Nedenstående tabel viser de 84 funktioner som IDEA 2002 indeholder. Med **fed** markerede funktioner behandles i dette modul. Det bemærkes at nogen af disse allerede er gennemgået.

<u>Karakter</u>		<u>Numerisk</u>	<u>Match</u>	<u>Dato / tid</u>	<u>Betinget</u>	<u>Finansiel</u>
@chr()	<b>@remove()</b>	<b>@abs()</b>	@between()	<b>@age()</b>	@compif()	@db()
@comparenocase()	@repeat()	@ascii()	@bit()	<b>@ctod()</b>	<b>@if()</b>	@ddb()
@curform()	<b>@replace()</b>	@curval()	@bitand()	@date()		@fv()
<b>@delete()</b>	@reverse()	@exp()	@bitor()	<b>@day()</b>		@ipmt()
@findoneof()	<b>@right()</b>	@int()	@list()	@daystod()		@irr()
@getat()	@soundex()	@log()	@match()	@dow()		@mirr()
<b>@insert()</b>	@soundlike()	@log10()	@nomatch()	@dtoc()		@npv()
@isblank()	@spacestoone()	@max()		@dtodays()		@pmt()
<b>@isin()</b>	@spanexcluding()	@min()		@dtoj()		@ppmt()
@isini()	@spanincluding()	@precno()		@jtod()		<b>@pv()</b>
@left()	@str()	@random()		@month()		@rate()
@len()	<b>@strip()</b>	<b>@round()</b>		@ntod()		@sln()
<b>@lower()</b>	@trim()	@recno()		@time()		@syd()
@ltrim()	@upper()	@seed()		@year()		
@mid()		@sqrt()				
@proper()		@stratum()				
<b>@regexpr()</b>		<b>@val()</b>				

**Anbefaling 2:** Se syntaksforklaringen i formeeditoren sideløbende med nedenstående forklaring af udvalgte @funktioner for at opnå fortrolighed med syntaksforklaringerne.

#### **@abs()**

@abs anvendes når en numerisk værdi skal betragtes som en absolut værdi. Et eksempel på dette er et udtræk, hvor man ønsker alle poster (debit som kredit) over et bestemt beløb f.eks. 10.000. I nogen tilfælde vil kreditbeløb stå med negativt fortegn, hvorfor kriteriet NUMERISKFELT > 10.000 vil udelukke negative værdier mindre end – 10.000. I stedet for at benytte formlen:

$$\text{NUMERISKFELT} > 10.000 \text{ .OR. NUMERISKFELT} < -10.000$$

kan @abs anvendes, så udtrykket er noget kortere. Dvs.:

$$\text{@abs(NUMERISKFELT)} > 10.000$$

#### **@if()**

@if består af en betingelse. Er betingelsen sand returneres en værdi eller en tekst. Hvis betingelsen er falsk returneres en anden værdi eller tekst. I nedenstående eksempel returneres JA hvis @abs(NUMERISKFELT) > 10.000 ellers returneres NEJ:

$$\text{@if(@abs(NUMERISKFELT)} > 10.000; \text{"JA"}; \text{"NEJ"})$$

Bemærk at betingelsen i pågældende eksempel er en formel i sig selv. Betingelsen kan være både numerisk, karaktermæssig eller bestå af en dato. En formel kan godt bestå af flere betingelser som i nedenstående eksempel:

$$\text{@if ( SELSKAB = "D" ; "Selskab D" ; @if(SELSKAB = "C";"Selskab C" ; @if(SELSKAB = "B";"Selskab B" ; "Selskab A" ) ) )$$

Dette kan selvfølgelig gøres mere elegant med: @insert(SELSKAB ; 1 ; "Selskab "), hvis feltet SELSKAB kun består af bogstaver mellem A og D.

### @day()

Funktionen tager et datofelt og returnerer en numerisk værdi svarende til dagen i måneden. F.eks. bliver "2000/11/02" til 2 ligesom "2001/10/02" og "1001/01/02" også returneres med værdien 2. Formlen er som følger:

### @day(DATOFELT)

Analogt med @day returnerer @month 11,10 og 01 og @year returnerer 2000, 2001 og 1001 i ovenstående eksempel.

### @isin()

Funktionen anvendes til at finde et eller flere bestemte tegn i et karakterfelt (is in). Findes tegnet eller tegnene, returnerer funktionen den position, hvor tegnet befinder sig eller tegnene begynder. Den returnerede værdi er derfor numerisk. Et eksempel er en søgning efter A/S i et karakterfelt, hvor A/S'er har speciel interesse. Det er vigtigt at bemærke, at funktionen er følsom overfor små og store bogstaver. Følgende formel:

### @ISIN("A/S", KARAKTERFELT)

vil derfor ikke medtage a/s. Skal der tages hensyn til både små og store bogstaver skal følgende formel anvendes:

### @ISINI("A/S", KARAKTERFELT)

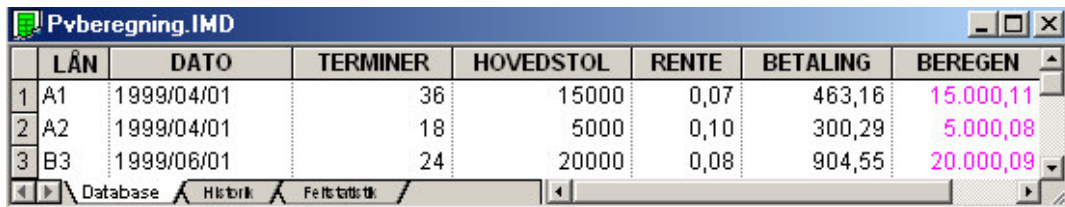
Formlen tager selvfølgelig ikke højde for slåfejl som AS eller lignende.

### @pv()

@pv beregner nutidsværdien ud fra oplysninger om antal terminer, rente og de periodemæssige betalinger. Formlen:

### @pv(BETALING ; RENTE/12 ; TERMINER)

beregner BEREGN svarende til den ønskede hovedstol som det kan ses nedenfor:



	LÅN	DATO	TERMINER	HOVEDSTOL	RENTE	BETALING	BEREGEN
1	A1	1999/04/01	36	15000	0,07	463,16	15.000,11
2	A2	1999/04/01	18	5000	0,10	300,29	5.000,08
3	B3	1999/06/01	24	20000	0,08	904,55	20.000,09

### @strip()

Funktionen er utrolig effektiv, idet den tager al "luften" ud af et karakterfelt. I næste afsnit vises et konkret eksempel på funktionens anvendelse. Funktionen er illustreret i nedenstående tabel:

Karakterfelt	@strip(Karakterfelt)
M e l l e m r u m	Mellemrum
Også 0 1 2 3 4 som tegn	Også01234somtegn

**@round()**

Funktionen @round anvendes til at afrunde værdier i et numerisk felt til heltal. Ved @(32,4) fås 32 og ved @(24,8) fås 25.

**@lower()**

Denne funktion er meget anvendelig, når der søges efter en bestemt tekst der kan være skrevet med både små og store bogstaver. Ved at konvertere store bogstaver til små bogstaver (eller omvendt vha. @upper) undgås fejlkilder. Et eksempel er vist nedenfor:

Karakterfelt	@lower(Karakterfelt)
FAKTURA 3451	faktura 3451
faktura 3452	faktura 3452
Faktura 3453	faktura 3453

Ønsker man derefter at søge efter teksten *faktura* er det betydelig lettere end hvis man skal søge på forskellige skrivekombinationer.

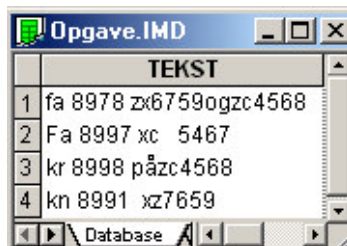
**@val()**

Ofte er talværdier defineret som karakterfelter. Det ses enten i Feltmanipulation eller ved at talværdierne er venstrestillet i databasevinduet. Konvertering til numeriske værdier gøres ved at anvende funktionen @val().

**4. Opbygning af komplekse formler**

For at opbygge mere komplekse formler, der indeholder adskillige @funktioner, er det hensigtsmæssig at opbygge formlen trin for trin.

Antag at opgaven er at foretage en numerisk hulopdagelse på et karakterfelt, der indeholder den nødvendige information. Desværre indeholder tekstfeltet også en del overflødig information, der vanskeliggør opgaven. Et udsnit af databasen ser således ud:



	TEKST
1	fa 8978 zx6759ogzc4568
2	Fa 8997 xc 5467
3	kr 8998 påzc4568
4	kn 8991 xz7659

Der er oplyst følgende om informationen:

- fakturanummer er forkortet **fa**
- kreditnota har to forskellige forkortelser **kr** og **kn**
- der ud over er der information om varetypen f.eks. zx6759

**Anbefaling 3:** opbygning af komplekse formler løses hurtigst ved at opbygge en formel trin for trin.

**Trin 1:** Det er hensigtsmæssigt at fjerne alle blanke tegn. Dette gøres ved @strip. Før formlen udbygges, kontrolleres det ønskede resultat.

@strip(tekst) giver resultatet:

	TEKST	STRIP
1	fa 8978 zx6759ogzc4568	fa8978zx67
2	Fa 8997 xc 5467	Fa8997xc54
3	kr 8998 påzc4568	kr8998påzc
4	kn 8991 xz7659	kn8991xz76

**Trin 2:** Efter at have konstateret det forventede i feltet STRIP kan man gå videre. Dette sker ved at bygge videre på @strip(tekst) ved at tilføje en funktion "uden på". Den valgte formel fjerner alle tegn, som ikke står 6 positioner fra venstre. Formlen udvides dermed til:

**@left(@strip(tekst);6).**

Allerede i trin 1 er det konstateret, at @strip(tekst) er korrekt, så den del skal ikke kontrolleres igen. Resultatet af @left(@strip(tekst);6) er:

	TEKST	VENTRE
1	fa 8978 zx6759ogzc4568	fa8978
2	Fa 8997 xc 5467	Fa8997
3	kr 8998 påzc4568	kr8998
4	kn 8991 xz7659	kn8991

**Trin 3:** Trin 3 minder om trin 2. De to første 2 tegn fjernes ved at beholde de 4 tegn fra højre. Det ordner formelen:

**@right(@left(@strip(tekst);6);4).**

**Trin 4:** Efter trin 3 fås den ønskede information, idet den dog er defineret som en karakter. Da mangler således endnu et led "uden på". Funktionen @val konverterer karaktertegn til numeriske værdier. Den endelige formel er derfor:

**@val(@right(@left(@strip(tekst);6);4))**

Pointen er, at man kan spare megen tid ved at opbygge sine formler trin for trin. Risikoen for at miste overblikket under opbygningen minimeres og undervejs kan formelen kontrolleres.

Det bemærkes, at den endelige løsning kan udformes på mange måder. Om formelen er "smart" eller mere omstændig er principielt underordnet, så længe den tjener sit formål.

Hvad nu, hvis det senere oplyses at "kr" følger sit eget nummersystem, hvorfor hulopdagelsen kun skal foretages på faktura og kreditnota med betegnelsen "kn"? Løsningen er givet her:

**@if(tekst = "kr";0;@val(@right(@left(@strip(tekst);6 );4)))**

I de fleste tilfælde vil et felt ikke indeholde en informationsstruktur, der kræver mere komplekse formler end ovenstående eksempel. Ofte kan det gøres med 1-3 trin.

## 5. Introduktion til @regexpr

Er informationsstrukturen alligevel mere kompleks og umiddelbart umulig at arbejde med, kan @regexpr sandsynligvis benyttes. I dette afsnit vises et eksempel på et regulært udtryk.

Antag at et tekstfelt indeholder information om, hvilke hjemmesider en række personer besøger. Informationen stammer oprindeligt fra en logfil. Opgaven er at rapportere, hvilke hjemmesider (hovedsiden), der besøges mest. En del af databasen og det ønskede resultat ser således ud:

Et tekstfelt i databasen	Det ønskede resultat
http://www.revisorinformatik.dk/Demonstration.htm	revisorinformatik.dk
www.jyskenetbank.dk/	jyskenetbank.dk
http://www.dr.dk/videnskab/?topbar=true	dr.dk
http://www.caseware-idea.com/fsr.asp?surl=%2Fsolutions%2Fdefault%2Easp	caseware-idea.com
www.nnerhverv.dk/login.asp	nnerhverv.dk

En løsning – ved anvendelsen af regulær udtryk - er:

```
@regexpr(HJEMMESIDER ; "[^http://www.][a-åA-Å.-]+")
```

Formlens første [ ] indikerer, at udtrykket eller dele af udtrykket i første [ ] skal findes. Det lille tegn ^ indikerer at udtrykket i sig selv ikke skal medtages. Anden [ ] indikerer, at kombinationer af små og stor bogstaver fra a til å skal medtages. Desuden skal . og - også medtages. Punktummet sikrer dermed, at .dk og .com etc. er med. Tegnet + sikrer, at alle tegn i [ ] før backslaceh (/) medtages. Resten følger af syntaksen til @regexpr.

En anden løsning - uden anvendelse af regulær udtryk - er:

```
@mid( @mid(@if(@isin("http://" ; HJEMMESIDER) =1; @mid(HJEMMESIDER ; 8 ; 50) ; HJEMMESIDER ) ;5;50); 1;@isin("");@mid(@if(@isin("http://" ; HJEMMESIDER) = 1; @mid(HJEMMESIDER ; 8 ; 50) ; HJEMMESIDER ) ;5;50))-1)
```

Den trinvis forklaring af formelen ses Appendix G.

Spørgsmålet er, hvilken teknik, det er hensigtsmæssigt for den enkelte bruger at tilegne sig. Regulær udtryk er hurtige og korte men er mere tidskrævende at lære. For yderligere litteratur om emnet kan der henvises til: "Mastering Regular Expressions", 2. udg. af Jeffrey, E. F. Friedl. Den findes på [www.amazon.com](http://www.amazon.com).

## 6. Udvalgelse af @funktioner

Den trinvis fremgangsmåde kræver udover teknikken en ide om, hvorledes et felt kan reduceres. Den viden, der er præsenteret i afsnittet "Ofte anvendte funktioner" er

tilstrækkelig til at få inspiration til hvilke formler, der kan være anvendelige i en given opgave. Det er erfaringen med tidligere formler, der letter opbygningen af nye formler.

**Anbefaling 4:** Begynd med at vælge den relevante funktionstype og søg derefter hjælp i syntaksforklaringerne eller i hjælpesystemet.

Lad os antage at man stilles overfor den problemstilling, at et numerisk felt skal konverteres til et karaktermæssigt felt. Det pågældende felt indeholder decimaltal.

Den relevante funktionsgruppe at søge i er gruppen Karakter. Derefter er der som regel ikke andet at gøre end at begynde fra en ende af. Fordelen her er den læringsmæssige sidegevinst ved at tage stilling til flere formlers anvendelighed.

De første to formler under Karakter er @chr og @comparenocase, hvilke kan udelukkes når syntaksforklaringerne betragtes. Derefter følger @curform og den synes brugbar. Bemærk at der ikke altid kun er én rigtig formel.

Derefter kan man forsøge sig lidt frem med udgangspunkt i syntaksforklaringen. Går det ikke og man stadig er sikker på at man har fat i en anvendelig funktion er der hjælp at hente i hjælpesystemet.



### Handling

- Åben formeledatoren.
- Klik på ikonet for "hjælp".
- Vælg @funktioner og derefter karakter samt @curform.
- Læs forklaringen på funktionen.

*Hjælp til formeledatoren*

**@CURFORM(Nudtr1;Sudtr1;Sudtr2;Nudtr2;Nudtr3)**

Hvor: *Nudtr1* = det felt, som reformateres

*Sudtr1* = tusindtalsseparator

*Sudtr2* = decimalseparator

*Nudtr2* = maksimumbredde for resultat

*Nudtr3* = antal decimaler

@CURFORM() udfører det modsatte af @CURVAL() og anvendes til at formatere et numerisk felt, med eventuelle tusindtals- og decimalseparatorer. Denne funktion returnerer et karakterfelt og bruges ofte til at reformatere et valutafelt, før resultatet fremkommer.

**Eksempel:**

@CURFORM(100234.78; " "; ", "; 12;2) vil returnere 100 234,78

@CURFORM(VÆRDI, " "; ", "; 12;2), hvor VÆRDI er et numerisk felt, vil returnere følgende:

VÆRDI (N 12,2)	@curform(VÆRDI;" ";",";12;2) (Virtuel karakter, Læn 14)
100456.45	100 456,45
456.00	456,00

## Appendiks G

Trinvis forklaring til følgende formel:

```
@mid( @mid(@if(@isin("http://" ; HJEMMESIDER) =1; @mid(HJEMMESIDER ; 8 ; 50) ; HJEMMESIDER ) ;5;50); 1;@isin("/") ;@mid(@if(@isin("http://" ; HJEMMESIDER) = 1; @mid(HJEMMESIDER ; 8 ; 50) ; HJEMMESIDER ) ;5;50))-1)
```

Før trin 1 ser feltet "hjemmesider" således ud:

HJEMMESIDER	
1	http://www.revisorinformatik.dk/Demonstration.htm
2	www.jyskenetbank.dk/
3	http://www.dr.dk/videnskab/?topbar=true
4	http://www.caseware-idea.com/fsr.asp?surl=%2Fsolutions%2Fdefault%2Easp
5	www.nnerhverv.dk/login.asp

**Trin 1:** Vha. funktionen @isin("http://"; HJEMMESIDER) fås en værdi af positionen af "http://". Hvis "http://" ikke indgår i en post returneres nul. Resultatet er følgende:

HJEMMESIDER		TRIN_1
1	http://www.revisorinformatik.dk/Demonstration.htm	1
2	www.jyskenetbank.dk/	0
3	http://www.dr.dk/videnskab/?topbar=true	1
4	http://www.caseware-idea.com/fsr.asp?surl=%2Fsolutions%2Fdefault%2Easp	1
5	www.nnerhverv.dk/login.asp	0

**Trin 2:** Værdien i feltet i POSITION kan anvendes til at fjerne "http://" med funktionen @mid. De første 7 tegn skal fjernes, hvis TRIN\_1 = 1. Den aktuelle formel er derfor @if(TRIN\_1 = 1; @mid(HJEMMESIDER ; 8 ; 50); HJEMMESIDER ). Det bemærkes at 8 i @mid er sat én højere end de 7 tegn i "http://". De 50 er sat højt for ikke at afskære væsentlig information. Resultatet er følgende:

HJEMMESIDER		TRIN_2
1	http://www.revisorinformatik.dk/Demonstration.htm	www.revisorinformatik.dk/Demon
2	www.jyskenetbank.dk/	www.jyskenetbank.dk/
3	http://www.dr.dk/videnskab/?topbar=true	www.dr.dk/videnskab/?topbar=tr
4	http://www.caseware-idea.com/fsr.asp?surl=%2Fsoluti...	www.caseware-idea.com/fsr.asp?
5	www.nnerhverv.dk/login.asp	www.nnerhverv.dk/login.asp

**Trin 3:** Når "http://" er væk ses det nu at de første 4 tegn i alle poster skal fjernes. Det gøres også med @mid. Dvs. @mid(trin 2; 4; 50). Resultatet er følgende:

	HJEMMESIDER	TRIN_3
1	http://www.revisorinformatik.dk/Demonstration.htm	revisorinformatik.dk/Demonstra
2	www.jyskenetbank.dk/	jyskenetbank.dk/
3	http://www.dr.dk/videnskab/?topbar=true	dr.dk/videnskab/?topbar=true
4	http://www.caseware-idea.com/fsr.asp?surl=%2Fsoluti...	caseware-idea.com/fsr.asp?surl
5	www.nnerhverv.dk/login.asp	nnerhverv.dk/login.asp

**Trin 4:** Nu mangler der en afskæring til højre dvs. formlen @mid kan anvendes igen. Sætter man @mid til startposition 1 og medtager 2 tegn vha. formlen @mid( @mid(Trin 3; 1;2) fås:

	HJEMMESIDER	TRIN_4
1	http://www.revisorinformatik.dk/Demonstration.htm	re
2	www.jyskenetbank.dk/	ly
3	http://www.dr.dk/videnskab/?topbar=true	dr
4	http://www.caseware-idea.com/fsr.asp?surl=%2Fsoluti...	ca
5	www.nnerhverv.dk/login.asp	nn

**Trin 5:** Dette trin skal ses i forlængelse af trin 4, idet vi ikke kun ønsker to tegn. Derfor skal der dannes en formel, der kan finde positionen med den første forekomst af "/". Den aktuelle formel er @isin("/"; trin 3; 5;50)-1. Minus 1 fordi "/" også skal afskæres. Benyttes @isin("/"; trin 3; 5;50) fås:

	HJEMMESIDER	TRIN_5
1	http://www.revisorinformatik.dk/Demonstration.htm	20
2	www.jyskenetbank.dk/	15
3	http://www.dr.dk/videnskab/?topbar=true	5
4	http://www.caseware-idea.com/fsr.asp?surl=%2Fsoluti...	17
5	www.nnerhverv.dk/login.asp	12

Den endelig formel består af trin 4 og 5 og resultatet fremstår som forventet:

	HJEMMESIDER	ENDELIG
1	http://www.revisorinformatik.dk/Demonstration.htm	revisorinformatik.dk
2	www.jyskenetbank.dk/	jyskenetbank.dk
3	http://www.dr.dk/videnskab/?topbar=true	dr.dk
4	http://www.caseware-idea.com/fsr.asp?surl=%2Fsoluti...	caseware-idea.com
5	www.nnerhverv.dk/login.asp	nnerhverv.dk