Databaser

Karen Asferg

2015

Århusvej 49 - 51
8500 Grenaa

Indholdsfortegnelse

Indledning

Hvorfor skal man arbejde med databaser? Og hvad kan de bruges til? Dette er nogle af de ting, der vil blive set på i det næste stykke tid. Vi vil se på databasetyper og især på den relationelle database, da det er den vi skal bruge sammen med vores websites. Vi vil også se lidt på normalisering af en database, så vi får så god en databasestruktur som muligt. Vi vil her bruge MSSQL, MSSQLEXPRESS og MySQL som eksempler på databaseservere og vi vil bruge HeidiSQL til MySQL til at få et brugervenligt interface, når vi arbejder direkte i databasen.

Vi vil også se på, hvordan en database bygges op og hvilket sprog, databaser snakker. Det hele krydres med eksempler og opgaver. Databaseundervisningen hænger jo nøje sammen med ASP.NET-undervisningen, så vi ser også på, hvordan man får lavet forbindelse mellem sin database og websitet.

Hvad er en database?

En database er et datalager. Dette datalager er en struktureret samling af dataenheder, som er gemt i en computer.

Databasens struktur er organiseret i henhold til en bestemt databasemodel f. eks. en relationel, som er den type, vi skal bruge, eller en hierarkisk eller en netværksmodel. Selve databasemodellen er generelt ikke noget vi beskæftiger os med, da den mest anvendelige og mest brugte model i dag er den relationelle.

Et af formålene med en database er at kunne organisere og strukturere information (dvs. dataenheder), så flere programtråde kan få adgang til denne type information på samme tid.

En relationel database har desuden den fordel, at man ud over at kunne gemme og manipulere klynger af data også kan gemme og manipulere relationer mellem den gemte data.

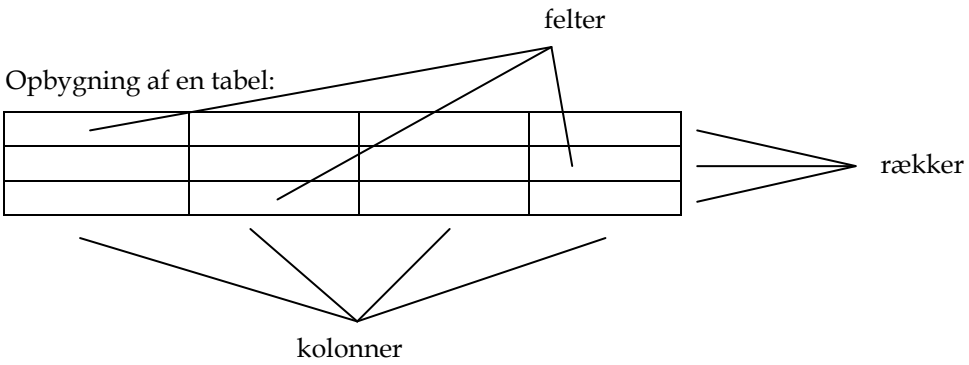
De data, der gemmes i en database kan stort set være af en hvilken som helst type (billeder, video, dokumenter osv.), men de mest almindelige datatyper i forbindelse med webudvikling er tekst og tal.

I en relationel database er data gemt i tabeller. Disse tabeller består af rækker, der igen består af felter. Hver række er en enhed. Alle rækker i en tabel består af det samme antal kolonner.

Dvs. at basisstrukturen i en relationel database er:

Tabeller
- Kolonner (fields)
- Rækker (posts)

Nøgler



I databaser bruges der et begreb, der kaldes nøgler. Der findes 2 typer af nøgler

Primærnøgler
Fremmednøgler

Primærnøgle: en kolonne (eller flere kolonner), hvis værdi er unik og som dermed identificerer den enkelte række

- En god primærnøgle har følgende karakteristika
- Den må ikke indeholde NULL-værdier eller være tom
- Den er unik (identificerer den enkelte række)
- Er stabil. Den ændres ikke
- Er kort. Indeholder ikke mange karakterer

Fremmednøgle: en kolonne i én tabel, der skal matche en kolonneværdi i en anden tabel. Hvis fremmednøglen ikke er NULL, skal der være et match i den tabel, der indeholder primærnøglen. Der laves en reference mellem to tabeller.

Udvikling af den relationelle database

Edgar Codd er den oprindelige udvikler af **R**elationelle **D**ata**B**ase**M**anagement **S**ystemer (RDBMS). Han publicerede i 1985 et sæt klare regler, som på daværende tidspunkt definerede idekomplekset bag et relationelt databasesystem.

Efterfølgende er disse regler blevet en del af den almene baggrundsviden inden for databaseverdenen. Brian Date har forklaret den relationelle model og gjort den almen tilgængelig og dermed øget forståelsen og brugen af det relationelle databasesystem.

Tidligere, dvs. for ca. 20 år siden, var det en kompleks sag at bygge en computerbaseret database. For at det kunne betale sig skulle databasen være kompleks og de eneste, der havde færdigheder, var professionelle databaseudviklere, for der var ingen brugervenlighed. Men tiderne har ændret sig med fremkomsten af Access, MySQL og MSSQL for blot at nævne nogle eksempler. Man har dermed fået software, som gør at det nu bliver enkelt for den almindelige bruger at lave og vedligeholde en database,

Data i en relationel database

Data i en relationel database skal som minimum overholde følgende kriterier:

Hver række skal have en unik nøgle

Alle rækker skal have samme antal kolonner

Alle felter, uanset datatype, skal have en fast længde. Selv såkaldte variable tekstfelter skal have en maksimal længde

Når man læser fra en database bruges tre grundlæggende operationer:

Filtrering: Man finder de rækker, der opfylder et bestemt kriterium f. eks. *navn='Hans'*

Visning: Vælg et antal kolonner fra en tabel, der skal indgå i resultatsættet. Det kunne f. eks. være fornavn, efternavn, alder

Krydsning/sammenføjning: Kombiner alle rækker i én tabel med alle rækker i en anden tabel

Databasetyper

Der findes mange forskellige databaser af forskellig kvalitet og kapacitet. De mest udbredte databaser er Oracle, IBM DB2, MySQL og MSSQL-Server. Oracle, IBM DB2 og MSSQL-server er kommercielle, mens MySQL er OpenSource.

De fire nævnte databaser er alle såkaldte service-baserede databaser, hvilket betyder, at de kører som en baggrundsservice på en server eller som hos os, når vi arbejder lokalt på en udviklingsmaskine. Denne service lytter kun efter forespørgsler til databasen på en specifik port, hvorefter den returnerer datasættet.

Af andre database-serverløsninger findes

- PostgreSQL
- SQL Server Compact Edition
- Sybase Advantage Database Server
- Sybase Adaptive Server Enterprise
- Informix
- Paradox
- Firebird
- AS/400(IBM iSeries)
- Progress
- Interbase
- Ingres
- Mimer SQL
- Lightbase
- Pervasive
- SQLBase
- Caché
- Teradata
- VistaDB
- DBMaker
- Netezza DBMS
- Valentina

Man kan som alternativ til en serverbaseret database også bruge en filbaseret database. Den kører så ikke på en service, men ligger gemt i en fysisk databasefil. Mange filbaserede databaser har et interface, der gør dem nemme at bruge til f. eks. generel kontorbrug.

Typiske filbaserede databaser er

- Access/Access 2007
- SQLite
- Visual FoxPro/FoxPro 2.x
- FileMaker

De filbaserede databaser er generelt ikke særlig velegnede til webudvikling, men kan have deres berettigelse ved desktopprogrammering

Foruden databaser findes der også simple datalagre, der ikke kan kaldes databaser, men som dog har nogle fællestræk med de filbaserede databaser. Disse er Excel /Excel 2007, XML og klartekstdokumenter (*.txt).

Disse kan bruges til at lagre og manipulere data, men man kan ikke lave avancerede forespørgsler mod dem.

Opbygning af databasen

Inden man kaster sig over computeren og begynder at lægge data ind i en database, skal man lave en analyse af de data, der skal være i databasen. Hvilke data skal der være og hvordan skal de hænge sammen. Hvor mange tabeller skal man oprette og hvilke data skal de forskellige tabeller indeholde? Hvor mange og hvilke kolonner skal den enkelte tabel indeholde? Hvilken datatype skal data i den enkelte kolonne have? Til dette brug skal man normalisere sin database.

Normalisering

At normalisere en database betyder at man overholder visse regler. Disse regler er blevet fastlagt af Codd. Ofte bruger man de tre første normalformer ud af de syv, der er. Dette vil i langt de fleste tilfælde være nok til at få en meget funktionel database.

De tre første normalformer er:

- Første normalform
- Anden normalform
- Tredje normalform

De fire yderligere normaliseringsniveauer er:

- BCNF(Boyce-Codd-Normalformen, der er en forstærkning af tredje normalform)
- Fjerde normalform (MVD: MultiValued Dependensies)
- Femte normalform (PJ/NF: Project-Join/Normal Form)
- DK/NF (Domain Key/Normal Form)

Ved at lave normalisering af en database undgår man redundant data, dvs. data, der gentages. Man minimerer også risikoen for fejl i data. Normalformerne skal sørge for at alle gentagne data lægges i separate tabeller, som man så relaterer til hinanden ved hjælp af nøgler. I praksis er dette ofte ikke nødvendigt, men man skal dog gerne normalisere til tredje grad.

Første normalform

En tabel er på første normalform, når:

- Der er en nøgle, der entydigt identificerer den enkelte række i tabellen.
- De enkelte felter i tabellen kun indeholder en værdi.
- Når der ikke er kolonner, der gentages

Anden normalform

En tabel er på anden normalform, når:

- Den opfylder alle kravene til første normalform.
- Ingen attributter/egenskaber, der ikke tilhører selve nøglen, må afhænge af en del af nøglen (alle kolonner i en tabel skal indeholde data om en og kun en entitet).

Tredje normalform

En tabel er på tredje normalform, når:

- Den opfylder alle kravene til anden normalform.
- Ingen attributter/egenskaber må afhænge af en del af andre attributter, der ikke selv er nøgler

Nedenstående tabel indeholder data om udlån af bøger

Låner	Bogdata	Bogdata	Dato for lån
Hanne Hansen Gyden 6 8500 Grenaa	Whitehorn, M. & Marklyn, B. Relationelle databaser IDG, 2003	Kanstrup, N. m. fl. Vildt & Landskab S & N, 2009	06-09-2009
Hanne Hansen Gyden 6 8500 Grenaa	Ingemarsson, K. Små gule citroner People's Press, 2009		06-09-2009
Hanne Jensen Nygade 6 8500 Grenaa	Kaaberbøl, L & Friis, A. Drengen i kufferten People's Press, 2008	Olsson, L. Astrid og Veronika Gyldendal, 2009	08-09-2009
Kirsten Jensen Borgergade 20 8410 Rønde	Wold, S. Fremkaldt Gyldendal, 2008	Kaaberbøl, L & Friis, A. Drengen i kufferten People's Press, 2008	08-09-2009
Malene Hansen Hovedgaden 4 8543 Hornslet	Jensen, J. F. Webdesign med stylesheets Globe, 2006		06-09-2009

Den er jo ikke særlig overskuelig og der står mange ting sammen. Der er også tomme felter.

Det, vi gerne vil, er at kunne registrere, hvad den enkelte har lånt og hvornår.

For det første kan man ikke identificere den enkelte bog. Drengen i kufferten findes i 2 eksemplarer og hvem har lånt hvilken? Dernæst har samme låner lånt flere bøger. Ikke alle lånere har lånt lige mange bøger.

Så tabellen skal normaliseres.

For at få denne tabel på første normalform skal tabellen opdeles i to mindre tabeller – en lånertabel med oplysninger om låneren og en udlånstabel med udlånsdata.

Lånertabel:

LånerID	Navn	Adresse	Postnummer	By
001	Hanne Hansen	Gyden 6	8500	Grenaa
002	Hanne Jensen	Nygade 6	8500	Grenaa
003	Kirsten Jensen	Borgergade 20	8410	Rønde
004	Malene Hansen	Hovedgaden 4	8543	Hornslet

Bogudlånsdata

BogID	LånerID	Forfatter	Titel	Forlag	Udgiv. år	Dato
1	001	Whitehorn, M. & Marklyn, B.	Relationelle databaser	IDG	2003	06-09-2009
2	001	Kanstrup, N. m. fl.	Vildt & Landskab	S & N	2009	06-09-2009
3	001	Ingemarsson, K.	Små gule citroner	People's Press	2009	06-09-2009
4	002	Kaaberbøl, L & Friis, A.	Drengen i kufferten	People's Press	2008	08-09-09
5	002	Olsson, L.	Astrid og Veronika	Gyldendal	2009	08-09-09
6	003	Wold, S.	Fremkaldt	Gyldendal	2008	08-09-09
7	003	Kaaberbøl, L & Friis, A.	Drengen i kufferten	People's Press	2008	08-09-09
8	004	Jensen, J. F.	Webdesign med stylesheets	Globe	2006	06-09-2009

Der står nu kun en oplysning i hvert felt og der er en unik nøgle til at identificere den enkelte række. Vi kan ved hjælp af brugerens Id se, hvem der har lånt hvilke bøger. Vi har nu opfyldt kravene til første normalform.

Ser vi på bogudlånstabellen kan vi se, at vore data om de enkelte bøger kun afhænger af den del af nøglen, der hedder BogId. Der er ingen direkte sammenhæng eller afhængighed med kolonnen LånerID. Vores kolonner indeholder derfor data om forskellige entiteter, idet udlån blandes sammen med data om den enkelte bog. Så vi opfylder ikke kravene til anden normalform, da der er kolonner, der ikke afhænger af nøglen. Det er et faktum her. Bogtitel, Forfatternavn, Udgivelsesår og Forlag afhænger kun af bogID. Det er uden videre muligt at udskifte et lånerID med et andet uden at det får konsekvenser for de nævnte kolonner. Man kan derimod ikke udskifte bogID uden at skifte de øvrige oplysninger.

Vi er derfor nødt til at oprette endnu en tabel, der kun handler om udlånet

UdlånsID	LånerID	BogID	Dato
1	001	1	06-09-2009
2	001	2	06-09-2009
3	001	3	06-09-2009
4	002	4	08-09-2009
5	002	5	08-09-2009
6	003	6	08-09-2009
7	003	7	08-09-2009
8	004	8	06-09-2009

Bogtabellen skal derfor også rettes til, så de ikke-tilhørende kolonner fjernes

BogID	Forfatter	Titel	Forlag	Udgiv. år
1	Whitehorn, M. & Marklyn, B.	Relationelle databaser	IDG	2003
2	Kanstrup, N. m. fl.	Vildt & Landskab	S & N	2009
3	Ingemarsson, K.	Små gule citroner	People's Press	2009
4	Kaaberbøl, L & Friis, A.	Drengen i kufferten	People's Press	2008
5	Olsson, L.	Astrid og Veronika	Gyldendal	2009
6	Wold, S.	Fremkaldt	Gyldendal	2008
7	Kaaberbøl, L & Friis, A.	Drengen i kufferten	People's Press	2008
8	Jensen, J. F.	Webdesign med stylesheets	Globe	2006

Vi har nu fået splittet vores data op i flere små enheder, som kun handler om et emne. Dog skal vores udlånstabel have et unikt nøglefelt, så de enkelte udlån kan skelnes fra hinanden. Der er nu kommet relationer mellem de enkelte tabeller, nemlig de forskellige id-felter, så det er derfor stadig muligt at få de samme oplysninger, som stod i den oprindelige tabel

Men der er i vores lånertabel stadig kolonner, der har et særligt forhold til hinanden, nemlig postnummer og by. Disse to egenskaber skal derfor ud i deres egen tabel

Postnummer	By
8500	Grenaa
8543	Hornslet
8410	Rønde

Lånertabellen får derefter følgende udseende

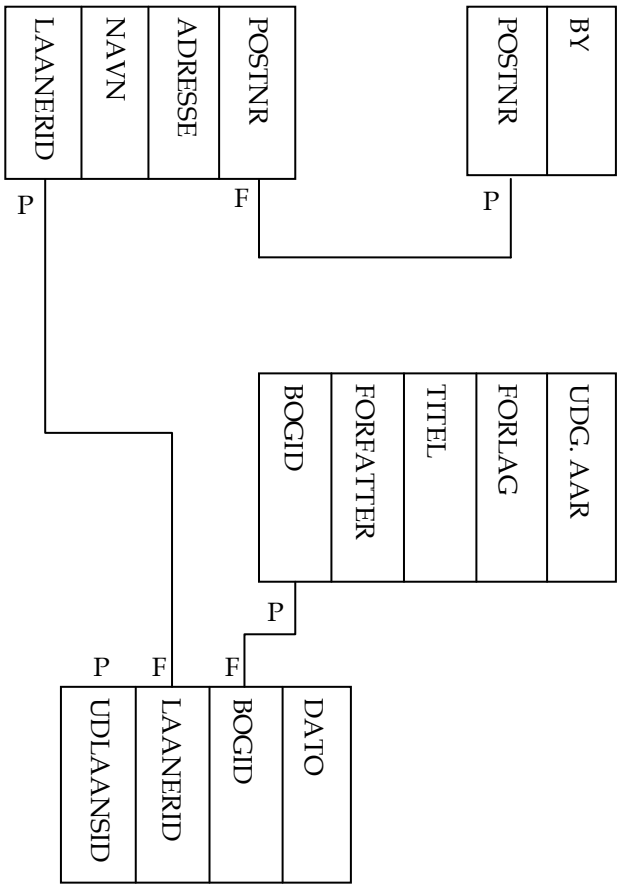
LånerID	Navn	Adresse	Postnummer
001	Hanne Hansen	Gyden 6	8500
002	Hanne Jensen	Nygade 6	8500
003	Kirsten Jensen	Borgergade 20	8410
004	Malene Hansen	Hovedgaden 4	8543

Så nu opfylder vores uoverskuelige tabel de tre første normalformer

Relationer

Selvom man har splittet sine data op i flere tabeller skal man jo stadig kunne få fat i sine data. Derfor skal der være relationer (dvs. forbindelser) mellem de forskellige tabeller. Der findes tre typer relationer

En til en
En til mange
Mange til mange



Opgave 1 (Normaliseringsopgave)

Nedenstående tabel skal normaliseres og afleveres

Hans Hansen	Stien 4	8000 Århus C	8633 3333	hans@hansen.dk	1 melon 19,-kr.	4 æbler 5 kr.	2 bananer 3, kr.
Jens Nielsen	Haven 5	8000 Århus C	8656 7654	jens@nielsen.dk	6 æbler 7,5 kr.		
Hanne Jensen	Broen 6	8500 Grenaa	8645 3456	hanne@jensen.dk	4 bananer 6 kr.	3 appelsiner 9 kr.	
Jette Hansen	Havstien 7	8410 Rønde	8737 3456	jette@hansen.dk	2 meleoner 38, kr	6 pærer 12,-kr	3 appelsiner 12,- kr.
Jytte Svendsen	Svinget 8	8543 Hornslet	8865 4577	jytte@svendsen.dk	8 æbler 10kr 8 pærer 16,- kr	4 appelseiner 12	
Poul Hansen	Bygmarken 5	8500 Grenaa	8765 3456	poul@svenden.dk	1 banan 1,25	1 banan 1,25	1 appelsin 6 kr.
Anne Borg	Markstien 8	8543 Thorsager	6756 3425	anne@borg.dk	2 meleoner 38	8 æbler 10 kr.	2 pærer 4 kr. 4 appelsiner
Jacob Jacobsen	Stigen 3	5800 Grenaa	4673 8262	jacob@jacobsen.dk	16 æble 20 kr.	10 pærer 20 kr.	
Jacob Jacobsen	Havblik 7	8410 Rønde	7898 7676		5 appelsiner 15 kr.		

Appelsin: 3,- kr.
Banan: 1,50 kr.
Melon: 19,- kr.
Pære: 2,- kr.
Æble: 1,25 kr.

Datatyper

Når man arbejder med databaser findes der nogle standard datatyper, som fortæller om de forskellige data. De mest brugte er til strengværdier og talværdier.

Tal

Når man arbejder med heltal skal man vælge en integer, der er stor nok til at indeholde de tal man skal bruge

Tekststrenge

Tekststrenge skal være af typen varchar(n). N'et angiver hvor mange karakterer der er afsat plads til.

Er det lange tekster kan man f. eks. vælge mediumtext

Tid

Til tidsangivelser findes der et specielt datoformat

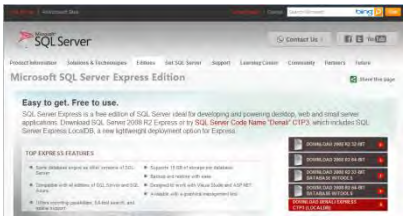
Penge

Float med angivelse af antallet af decimaler

Boolean

Er der kun to muligheder kan man bruge en boolean

Installering af MSSQLEXPRESS-server



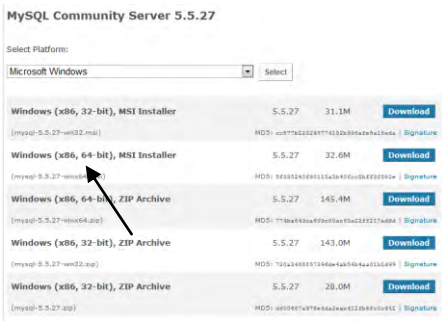
På http://www.microsoft.com/sqlserver/en/us/editions/previous-versions.aspx vælges menupunktet 2008 R2 Express i højre side. På den side, man kommer til vælges download SQLEXPR_x64_ENU.exe, hvis man har 64 bit styresystem. Ellers vælges til 32 bit.
Databasen installeres. På dette link er en tutorial til installationen. http://teamtutorials.com/windows-tutorials/installing-microsoft-sql-2008-express-edition

Klik på næste hele vejen igennem. Husk at angive et password og et servernavn. Og husk at vælge mixed mode.

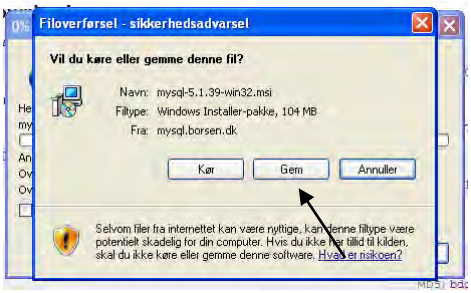
Installering af MySQL-server

Vi er nu nået så langt, at vi skal til at installere MySQL server.
På deres hjemmeside
http://dev.mysql.com/downloads/
vælges Community Server recommended

Når man har valgt dette punkt får man mulighed for at vælge en hel masse forskellige typer af serveren, alt efter styresystem.
Vi vælger Windows til 64 bit, hvis styresystemet er 64 bit. Ellers vælges 32 bit



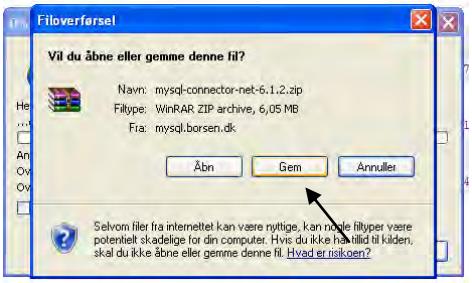
Når vi har valgt Windows skal vi igen tage stilling.
Her vælger vi Windows MSI Installer (x86, 64 bit).
Tryk på download
Når dette vindue kommer frem skal der trykkes på Gem.
Filen skal gemmes fysisk på computeren.
Når filen skal pakkes ud, skal udpakningen også gemmes fysisk på computeren. F. eks. på skrivebordet.



Dernæst vælges i højre side mySQLConnector/Connector/NET, som er forbindelsen mellem asp.net og serveren)

Her vælges Windows Binaries (zapped MSI Installer). Tryk på download

Når dette vindue kommer frem skal der trykkes på Gem.
Filen skal gemmes fysisk på computeren.
Når filen skal pakkes ud, skal udpakningen også gemmes fysisk på computeren. F. eks. på skrivebordet.



På hjemmesiden
http://www.heidisql.com/download.php
under download hentes Installer

Når alle tre filer er pakket ud, skal de installeres. Dobbeltklik på MySQL5.1xx.msi. Klik next hele vejen igennem, undtagen når der skal angives password. Angiv her et password som man helt sikkert kan huske.

Dobbeltklik derefter på MySql.Data.msi. Så bliver Visual Studio opdateret til også at kunne håndtere en MySQL-Server

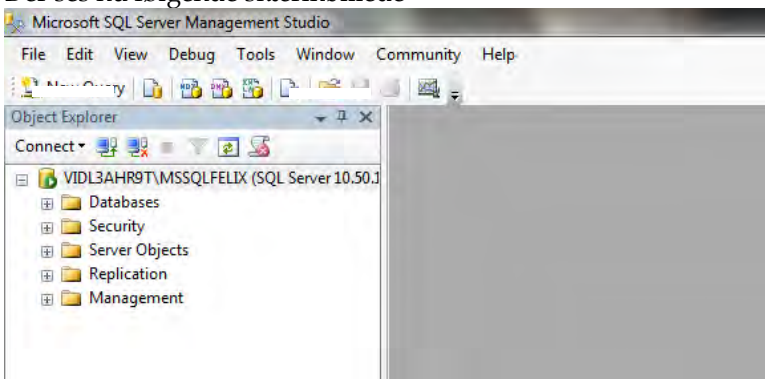
Til sidst dobbeltklikkes på HeidiSQL_4.0_Setup.exe. Brugerinterfacet HeidiMySQL bliver installeret

Servernavn skal være localhost.

Oprettelse af en MSSQL-database

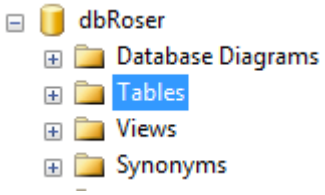
Nu er vi klar til at oprette en lille database.

Åbn SQL Server Management Studio og lav forbindelse ved at trykke på Connect
Der ses nu følgende skærmbillede



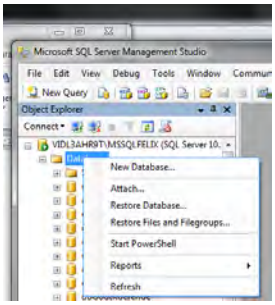
Klikker man på punktet Databases, kan her se de databaser, der er oprettet. System Databases har systemet selv lavet.

Klikker man på det lille plus ud for en database, her dbRoser, kan man se forskellige punkter, bl.a. punktet Tables. Højreklikker man på punktet Tables, så får man mulighed for at lave nye tabeller.



På billedet kan man desuden se, hvor mange tabeller, der er i den valgte database. Her Roser, som består af 2 tabeller.

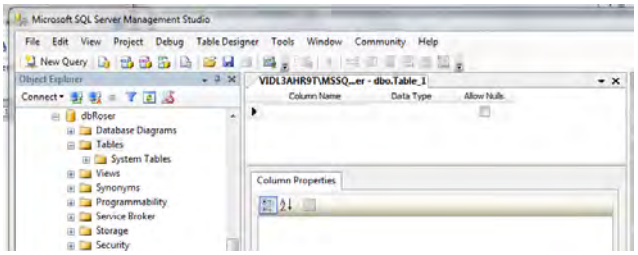
Ved at højreklikke på Databases i databaselisten får man mulighed for at oprette en ny database.



Vælg New Database.

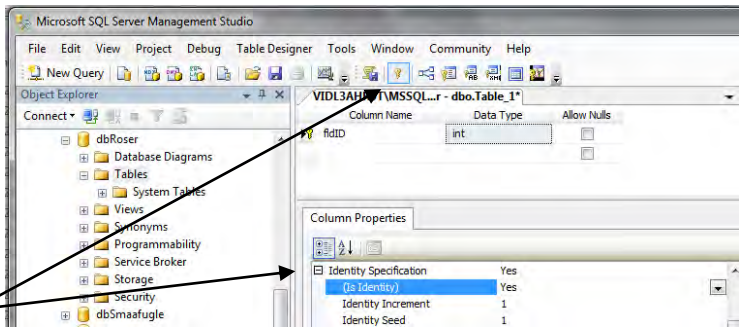
Kald databasen for dbRoser. Klik på OK

Klik på plusset ud for dbRoser og vælg højreklik her på punktet Tables. Vælg New Table. Der kommer nu følgende skærmbillede.



I kolonnen Column Name skal man skrive sit feltnavn.
Kolonnen Data Type skal indeholde datatypen og kolonnen Allow Nulls er en ja/nej kolonne.

Den første kolonne skal navngives fldID og være af datatypen int, som repræsenterer integers, dvs. heltal. Denne kolonne skal være den unikke id for den enkelte række.. Derfor skal det defineres, at det er en indentifikationskolonne og en primærnøgle. Disse egenskaber sætter man i Column Properties-vinduet.



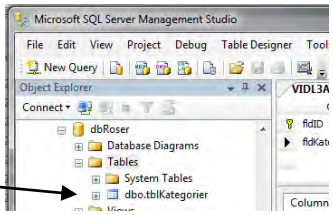
Scroll ned til punktet Identity Specification, klik på plusset og vælg Yes. Klik dernæst på nøglen.

Bemærk, at Null-værdier nu ikke længere er tilladt (Der er ikke længere flueben i afkrydsningsfeltet).

Bemærk også at fletet Identity Increment er sat til 1. Det betyder at autonummereringen stiger med og man skal IKKE selv skrive i fldID-feltet.

Sletter man en række, kan man ikke genbruge den slettede id-værdi.

Den næste kolonne skal hedde fldKategori og være af datatypen varchar. Tallet i parentes angiver, hvor mange karakterer feltet kan indeholde). Sæt det til 20 (Length i Column Properties). Gem tabellen ved at klikke på gemsymbolet eller Control + S. Giv tabellen navnet tblKategorier. Bemærk, at man nu kan se tabellen i højre side.

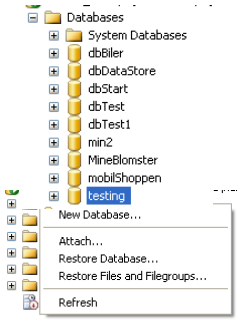


Der skal nu lægges data i tabellen. Højreklik på tabellen tblKategorier og vælg Edit top 200, så kommer man til stedet, hvor man kan lægge data ind i sin tabel (husk af feltet fldID bruger autonummerering).

Data i tabellen skal være

- Engelske roser
- Klatreroser
- Slyngroser

Oprettelse af endnu en MSSQL-database

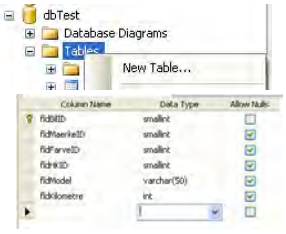


Der skal nu oprettes endnu en database, nemlig dbFrugt

Når man skal oprette en ny database, højreklikker man på Databases og vælger New Database.

Databasen navngives med et passende navn, som passer til emnet, f. eks. dbFrugt. Klik OK.

Dernæst skal der oprettes tabeller. Højreklik på Tables under databasenavnet og vælg New Table. Indtast kolonnenavne og giv dem en datatype passende til deres indhold.



Feltet fldID skal sættes til nøglefelt og være med autonummerering

Feltnavn	Datatype	Længde
fldID	smallInt	
fldVarenavn	varChar	20
fldPris	Float unsigned	
fldAntal	smallInt unsigned	5

Gem designet for tabellen og navngiv tabellen med et passende navn, der passer til de data den skal indeholde, f.eks tblFrugt. Højreklik derefter på tabelnavnet og vælg Edit Top 200 Rows. Læg data i tabellen (Se nedenfor).

fldID	fldVarenavn	fldPris	fldAntal
1	æbler	1,25	1000
2	pærer	2	2000
3	appelsiner	3	3000
4	meloner	19	500
5	bananer	1,5	1000

Navngivning i databaser:

Det er ikke nødvendigt, men en god ide at navngive sin database og dens indhold med noget, der passer til det projekt, man arbejder med. Ofte giver man tabeller og kolonner et præfix, så de er nemmere at se og identificere.

Databaser får præfikset db og derefter et navn, der passer til indholdet
Tabeller får præfikset tbl og derefter et navn, der passer til indholdet
Kolonner får præfikset fld og derefter et navn, der passer til indholdet

Databasesyntaks

Vi er nu klar til at lave nogle udtræk (forespørgsler) i vores tabel. Når man skal lave udtræk er der nogle reserverede ord, dvs. ord, som SQL'en kender og forbindes med specielle funktionaliteter. Dem vi skal kende i første omgang er

Ord	Spørgsmål	Resultat
SELECT	➢ Hvad returnerer SELECT? ➢ Må man skrive SQL uden SELECT? ➢ Hvad må SELECT indeholde?	Returnerer kolonner i resultatsættet Er der ingen SELECT er der intet resultatsæt at læse Man kan foretage beregninger i SELECT. SELECT kan indeholde SUB-SELECT
FROM	➢ Hvori indgår indholdet fra FROM? ➢ Hvad må FROM indeholde ➢ Må man skrive SQL uden FROM?	Er datagrundlaget for resultatsættet Indeholder datakilder: Udgangspunkt for næsten alle resultatsæt! Resultatsættet stammer altid: Enten direkte fra beregning i SELECT (sjældent) Eller fra indholdet i FROM (næsten altid) Kan bl.a. indeholde JOINS og SUB-SELECT
WHERE	➢ Hvad håndterer WHERE? ➢ Hvad indeholder WHERE? ➢ Hvad returnerer WHERE? ➢ Hvad er effekten af WHERE?	WHERE håndterer enkeltrækker WHERE indeholder et logisk udtryk WHERE returnerer true, false eller unknown Det er kun enkeltrækker, som returnerer true, der kommer videre til resultatsættet
ORDER BY	➢ Hvad gør ORDER BY? ➢ Hvad indeholder ORDER BY? ➢ Hvad håndterer ORDER BY? ➢ Hvornår bør ORDER BY bruges?	Sætter det endelige resultatsæt op efter nogle givne definitioner Indeholder en kommesepareret liste, som resultatsættet ordnes efter. Man kan angive om ens resultatsæt skal vises stigende (ASC er default) eller faldende (DESC) for de valgte elementer i listen Man bør kun bruge ORDER BY, hvis man skal se resultatsættet direkte
LIMIT (MySQL) TOP (MSSQL)	➢ Hvad håndterer LIMIT/TOP ➢ Hvad returnerer LIMIT/TOP ➢ Hvornår bruges LIMIT/TOP	LIMIT/TOP bruges til at begrænse et antal rækker taget fra starten eller slutningen af en tabel

Aggregatfunktioner

Hvad er en aggregatfunktion?

En aggregatfunktion er en funktion, der kan bruges til at lave beregninger med eller til at finde bestemte værdier.

Aggragatfunktioner er MIN, MAX, AVG, SUM, COUNT, COUNT DISTINCT

En aggregatfunktion skal have input fra en eller flere rækker. Dette input kan være tekst eller numerisk

SUM og AVG tager kun numerisk input

MIN, MAX og COUNT tager både tekst og numerisk input.

En aggregatfunktion returnerer altid én værdi

Operatorer

En operator giver mulighed for at udvælge data efter forskellige kriterier

Operatorer er bl.a. =, <, >, <=, >=, LIKE, NOT LIKE, AND, OR

Vi vil nu trække data ud af tabellen
Vi vil finde:
Alle varer vist med varenavn
SELECT fldVarenavn, fldPris FROM tblVarer

Alle varer vist med varenavn og pris sorteret efter varenavn
SELECT fldVarenavn, fldPris FROM tblVarer ORDER BY fldVarenavn

Alle varer vist med varenavn og pris sorteret faldende efter varenavn
SELECT fldVarenavn, fldPris FROM tblVarer ORDER BY fldVarenavn DESC

De varer, som koster 3 kroner eller mere. Varenavn og pris skal vises
SELECT fldVarenavn, fldPris FROM tblVarer WHERE fldPris >= 3 ORDER BY fldVarenavn

De varer, som koster 3 kroner eller mere. Varenavn og pris skal vises faldende
SELECT fldVarenavn, fldPris FROM tblVarer WHERE fldPris >= 3 ORDER BY fldVarenavn DESC

De varer, som koster mindre end 3 kroner. Varenavn og pris skal vises faldende
SELECT fldVarenavn, fldPris FROM tblVarer WHERE fldPris < 3 ORDER BY fldVarenavn DESC

Tæl, hvor mange varegrupper, der findes
SELECT COUNT(fldID) FROM tblVarer

Tæl, hvor mange varegrupper, der findes, som koster mindre end 3 kroner
SELECT COUNT(fldID) FROM tblVarer WHERE fldPris < 3

Find den højeste pris
SELECT MAX(fldPris) FROM tblVarer

Find den laveste pris
SELECT MIN(fldPris) FROM tblVarer

Find gennemsnitsprisen på varerne. Kolonnen skal vises med overskriften Gennemsnit
SELECT AVG(fldPris) 'Gennemsnit' FROM tblVarer

Find summen på varerne. Kolonnen skal vises som SUM
SELECT SUM(fldPris) as 'SUM' FROM tblVarer

Find de varegrupper, deres pris og antal, hvor antallet er større end 600 og mindre end 2000
SELECT fldVarenavn, fldPris, fldAntal FROM tblVarer WHERE fldAntal > 600 and fldAntal < 2000

Find den første vare
MSSQL
SELECT TOP 1 fldVarenavn, fldPris FROM tblVarer ORDER BY fldID

MySQL
SELECT fldVarenavn,fldPris FROM tblVarer ORDER BY fldID LIMIT 1

Find de 3 nyeste varer
MSSQL
SELECT TOP 3 fldVarenavn, fldPris FROM tblVarer ORDER BY fldID DESC

MySQL

SELECT fldVarenavn, fldPris FROM tblVarer ORDER BY fldID DESC LIMIT 3

Find de varer, hvis navn indeholder et a

SELECT fldVarenavn FROM tblVarer WHERE fldVarenavn LIKE '%a%'

Find de varer, hvis navn ikke indeholder et a

SELECT fldVarenavn FROM tblVarer WHERE fldVarenavn NOT LIKE '%a%'

Find de varegrupper, deres pris og antal, hvor antallet er større end 600 og mindre end 2000 og i hvis navn et a indgår

SELECT fldVarenavn, fldPris, fldAntal FROM tblvarer WHERE (fldAntal > 600 AND fldAntal < 2000) AND fldVarenavn LIKE '%a%'

Find de varegrupper, deres pris og antal, hvor antallet er større end 600 og mindre end 2000 og i hvis navn et a ikke indgår

SELECT fldVarenavn, fldPris, fldAntal FROM tblvarer WHERE (fldAntal > 600 AND fldAntal < 2000) AND fldVarenavn NOT LIKE '%a%'

Find alle varer, der har bogstaverne an som 2. og 3. bogstav

SELECT fldVarenavn FROM tblVarer WHERE fldVarenavn LIKE '_an%'

Find alle de varer, hvis varenavn begynder med *a* efterfulgt af et tilfældigt bogstav efterfulgt af *pel*, efterfulgt af et tilfældigt bogstav, efterfulgt af *in*

SELECT fldVarenavn FROM tblVarer WHERE fldVarenavn LIKE 'a_pel_in%'

Find varenavn, pris og antal grupperet efter antal

SELECT fldVarenavn, fldPris, fldAntal FROM tblVarer WHERE fldPris > 2 GROUP BY fldAntal, fldVarenavn, fldPris

Find ud af, hvor mange penge varelageret står i for hver enkelt vare

*SELECT fldVarenavn, fldPris, fldAntal, (fldAntal*fldPris) as 'SUM' FROM tblVarer*

Opgave 2

Opret nu selv følgende kundetabel, som skal indeholde oplysninger som fornavn, efternavn, adresse, postnummer, telefon og e-mail.

Fornavn	Efternavn	Adresse	Postnr	Telefon	e-mail
Hans	Hansen	Stien 4	8500	8637 9999	hanshansen@hansen.dk
Jonna	Jensen	Bygaden 4	8500	8634 5454	jonna@jensen.dk
Kirsten	Nielsen	Strædet 7	8543	8699 4444	kirsten@nielsen.dk
Jacob	Poulsen	Havnen 9	8410		
Jacob	Nielsen	Bygaden 4	8400	8675 4673	jacob@nielsen.dk
Karina	Poulsen	Stien 6	8500	8634 4343	karina@poulsen.dk
Jan	Poulsen	Hovedgaden 45	8400		
Marina	Johnsen	Hovedgaden 66	8400	8655 5555	marina@johnsen.dk
Carsten	Nielsen	Vænget 4	8410	8644 4444	carsten@nielsen.dk
Karsten	Mortensen	Rugmarken 3	8543	8633 3333	

Find derefter følgende om Jeres kunder:
Alle kunder skal vises med fornavn, efternavn og adresse

Alle kunder sorteret faldende på efternavn, fornavn

Antallet af kunder

Alle kunder, der bor indenfor postnumrene 8000 til 8410, begge inklusive

Alle kunder, der bor udenfor postnumrene 8000 til 8410

Alle de kunder, der ikke har @ eller . i deres mailadresse

I varetabellen skal I finde den samlede pris på de enkelte varegrupper

Find de varer, der koster mere end 1,50 kr. per stk.

Find også de varer i hvis navn et e indgår

Find også de varer i hvis navn et e ikke indgår

Tjek om der er kunder, der har et t som andet bogstav i efternavnet

Find de sidste 3 kunder

Find den første kunde

JOINS

Med JOINS får man adgang til data i flere tabeller på samme tid. Det vil sige, at JOIN sammenknytter rækker. Hvordan disse rækker bliver knyttet sammen afhænger af JOIN-typen.

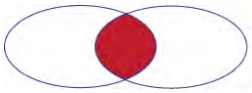
JOIN optræder kun i FROM-delen af et SQL-udtræk og før WHERE, ORDER BY osv.

INNER JOIN er de rækker, der har en kolonne fælles.
OUTER JOIN er de rækker, i den ene tabel, hvor der ikke er fællesskab med en kolonne i den anden tabel plus de rækker, hvor der er fællesskab mellem de to valgte kolonner.

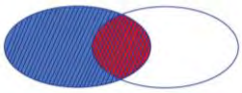
I en JOIN tager man to tabeller og sætter i forlængelse af hinanden, så man får adgang til alle kolonner i begge tabeller.

JOINS kan illustreres på følgende måde:

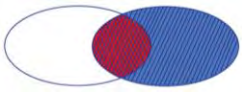
INNER JOIN:
tabel 1 INNER JOIN tabel 2



LEFT OUTER JOIN:
tabel 1 LEFT OUTER JOIN tabel 2



RIGHT OUTER JOIN:
tabel 1 RIGHT OUTER JOIN tabel 2



- Fællesmængden mellem to tabeller

Fællesmængden mellem to tabeller returnerer altid TRUE

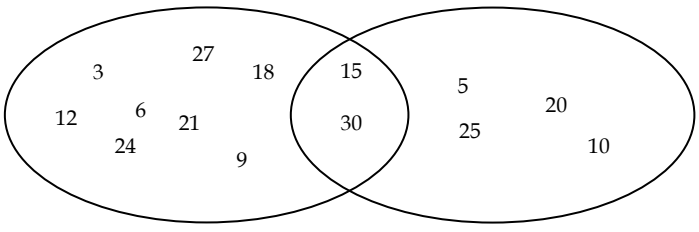
For at illustrere fællesmængder, vil vi her lave et par eksempler:

Vi har to talgrupper, nemlig en, hvor
alle tal til og med 30 der er delelige med 3
og en hvor
alle tal til og med 30, der er delelige med 5
befinder sig

Den ene mængde bliver
3 – 6 – 9 – 12 – 15 – 18 – 21 – 24 – 27 – 30

Den anden mængde bliver
5 – 10 – 15 – 20 – 25 – 30

I disse to mængder er der nogle fælles tal, nemlig 15 og 30.
Grafisk kan dette illustreres således:



Et andet eksempel kan være at finde fællesmængden mellem de to grupper kæledyr og rovdyr

Kæledyr:	**Rovdyr:**
Hund	Kat
Hest	Løve
Marsvin	Mår
Kanin	Hyæne
Kat	Hund

De to dyr, der går igen i begge grupper er kat og hund. Det er derfor disse to, som er fællesmængden her.

Endnu et lille eksempel
Vi har de to grupper danske ynglefugle og trækfugle i Danmark

Danske ynglefugle	**Trækfugle i Danmark**
Gråspurv	Havesanger
Landsvale	Munk
Bysvale	Landsvale
Solsort	Nøddekrige
Bogfinke	Kortnæbbet gås
Nattergal	Bysvale
Munk	Kvækerfinke
Havesanger	Snespurv
	Nattergal

De fugle, der findes i begge grupper er:
Landsvale
Bysvale
Havesanger
Munk
Nattergal

Det er derfor disse 5 arter, der er fællesmængden her

Inden vi for alvor går i gang med JOINS i databasen vil vi oprette endnu en tabel, nemlig en ordretabel, der styrer hvor mange ordrer, der er kommet til firmaet.

id	Vare-id	Kunde-id	Antal købte varer
1	1	1	10
2	2	1	5
3	1	2	10
4	3	2	10
5	5	6	15

Der findes som sagt tre typer af JOINS: LEFT, RIGHT og INNER Den vi bruger mest er INNER, hvor man får adgang til to (eller flere) tabellers data via to kolonner, en i hver tabel.
En INNER JOIN returnerer fællesmængden mellem to tabeller, dvs. præcis de enkeltrækker, hvor JOIN-klausulen returnerer TRUE.

En JOIN har altid et logisk udtryk

Fra vores to tabeller vil vi gerne se hvilke ordrer der er afgivet

Vi laver derfor følgende SQL-statement

SELECT tblordrer.fldID, tblordrer.fldKoebtAntal, tblkunder. FROM tblordrer, tblkunder WHERE tblordrer.fldKundeID=tblkunder.fldID*

Som giver følgende resultat

Ordre-ID	Antal købte varer	Kunde-ID	Fornavn	Efternavn	Adresse	postnr	Telefon	Email
1	10	1	Hans	Hansen	Stien 4	8500	8637 9999	hanshansen@hansen.dk
2	5	1	Hans	Hansen	Stien 4	8500	8637 9999	hanshansen@hansen.dk
3	10	2	Jonna	Jensen	Bygaden 4	8500	8634 5454	jonna@jense.dk
4	10	2	Jonna	Jensen	Bygaden 4	8500	8634 5454	jonna@jense.dk
5	15	6	Karina	Poulsen	Stien 6	8500	8634 4343	karina@poulsen.dk

Dette er helt det samme som

SELECT tblordrer.fldID, tblordrer.fldKoebtAntal, tblkunder.
FROM tblordrer INNER JOIN tblkunder ON tblordrer.fldKundeID=tblkunder.fldID

Man kan selvfølgelig sætte betingelser på en JOIN i en WHERE-klausul.

Find de kunder, der har opgivet et telefonnummer

SELECT fldForNavn,fldTelefon FROM tblKunder WHERE fldTelefon is NOT NULL

Vis hvor mange af de forskellige varer kunderne har købt. Der skal vises fornavn, varenavn og antal

SELECT k.fldForNavn, v.fldvarenavn, o.fldKoebtAntal AS 'Antal' FROM tblKunder AS k INNER JOIN tblOrdrer AS o ON k.fldID=o.fldKundeID INNER JOIN tblVarer AS v ON o.fldVareID=v.fldID

Vis ordrerne på de kunder, der har købt over 6 stk. af en vare. Der skal vises fornavn, varenavn og antal

SELECT k.fldForNavn, v.fldvarenavn, o.fldKoebtAntal AS 'Antal' FROM tblKunder AS k INNER JOIN tblOrdrer AS o ON k.fldID=o.fldKundeID INNER JOIN tblVarer AS v ON o.fldVareID=v.fldID WHERE o.fldKoebtAntal>6 ORDER BY k.fldFornavn

OUTER JOIN: Her tager man alle rækker fra den ene tabel, mens man fra den anden tager rækker, der opfylder et eller kriterier. OUTER JOINS kan være left eller right. En OUTER JOIN vil f. eks. kunne bruges til at finde de kunder, der ikke har afgivet en ordre eller til at finde varer, som ingen kunder har købt.

Hvis vi vil have en liste over alle kunder og deres ordrer, uanset om de har købet noget eller ej, så skal man tage udgangspunkt i tabellen tblKunder, da vi jo gerne vil have vist alle vore kunder.

SQL-statementet bliver derfor

SELECT tblkunder.fldFornavn, tblkunder.fldEfternavn, tblkunder.fldAdresse, tblkunder.fldPostnr, tblordrer.fldVareID, tblordrer.fldKoebtAntal
FROM tblKunder LEFT OUTER JOIN tblordrer ON tblKunder.fldID=tblordrer.fldKundeID

Og tabellen får dette indhold:

Fornavn	Efternavn	Adresse	Postnr	Vare-ID	Købt antal
Hans	Hansen	Stien 4	8500	1	10
Hans	Hansen	Stien 4	8500	2	5
Jonna	Jensen	Bygaden 4	8500	1	10
Jonna	Jensen	Bygaden 4	8500	3	10
Kirsten	Nielsen	Strædet 7	8543	NULL	NULL
Jacob	Poulsen	Havnen 9	8410	NULL	NULL
Jacob	Nielsen	Bygaden 4	8400	NULL	NULL
Karina	Poulsen	Stien 6	8500	5	15
Jan	Poulsen	Hovedgaden 45	8400	NULL	NULL
Marina	Johnsen	Hovedgaden 66	8400	NULL	NULL
Carsten	Nielsen	Vænget 4	8410	NULL	NULL
Karsten	Mortensen	Rugmarken 3	8543	NULL	NULL

Det bemærkes, at hvis en kunde ikke har afgivet en ordre, så returnerer SQL NULL
Dette kan vi udnytte, hvis vi gerne vil finde de kunder, der ikke har afgivet en ordre.

```
SELECT tblkunder.fldFornavn, tblkunder.fldEfternavn, tblkunder.fldAdresse, tblkunder.fldPostnr
FROM tblKunder LEFT OUTER JOIN tblordrer ON tblKunder.fldID=tblordrer.fldKundeID WHERE
tblordrer.fldID IS NULL
```

Fornavn	Efternavn	Adresse	Postnr
Kirsten	Nielsen	Strædet 7	8543
Jacob	Poulsen	Havnen 9	8410
Jacob	Nielsen	Bygaden 4	8400
Jan	Poulsen	Hovedgaden 45	8400
Marina	Johnsen	Hovedgaden 66	8400
Carsten	Nielsen	Vænget 4	8410
Karsten	Mortensen	Rugmarken 3	8543

Man kan selvfølgelig også tælle, hvor mange kunder, der ikke har afgivet en ordre.

```
SELECT COUNT(*) AS 'Antal kunder uden ordre' FROM tblkunder LEFT OUTER JOIN tblordrer
ON tblkunder.fldID=tblordrer.fldKundeID WHERE tblordrer.fldID IS NULL
```

Forskellen på en left og en right join er, at ved en left join er det den venstre tabel i udtrykket, der er intakt, mens det ved en right join er den højre. Dvs. det er rækkefølgen, som tabellerne listes i, der bestemmer om man skal bruge en left join eller en right join.

Prøv selv at lave ovenstående SQL-statement om til en right join

```
SELECT COUNT(*) AS 'Antal kunder uden ordre' FROM tblordrer RIGHT OUTER JOIN tblkunder
 ON tblkunder.fldID=tblordrer.fldKundeID WHERE tblordrer.fldID IS NULL
```

GROUP BY

Med GROUP BY får man mulighed for at udvide SELECT-forespørgslen med en gruppering af data. GROUP BY virker IKKE på samme måde som ORDER BY. ORDER BY styrer den rækkefølge, som data skal returneres i, mens GROUP BY inddeler de fundne data i grupper, som derefter bliver behandlet som én enhed af databasesystemet. Princippet er, at alle de rækker, der har identiske værdier i den kolonne, der grupperes efter, bliver slået sammen og derfor bliver returneret af SQL som en række med sammentællinger, der er baseret på de rækker, der indgår i gruppen.

GROUP BY bruges sammen med forskellige gruppefunktioner som COUNT(), AVG(), MAX(), MIN(), SUM().

Hvis man f. eks. vil se, hvor mange kunder der bor inden for de forskellige postnumre, kan man bruge en GROUP BY.

```
SELECT fldPostnr, COUNT(*) AS 'Antal kunder' FROM tblkunder
 GROUP BY fldPostnr
```

Dette SQL-statement giver følgende resultat

Postnummer	**Antal kunder bosiddende i valgte postnummer**
8400	3
8410	2
8500	3
8543	2

GROUP BY kan også bruges til at se, hvor mange ordrer kunderne har afgivet.

```
SELECT fldKundeID, COUNT(*) AS 'Antal ordrer' FROM tblordrer
 GROUP BY fldKundeID
```

Dette SQL-statement giver følgende resultat

KundeID	**Antal ordrer**
1	2
2	2
6	1

Ønsker man en liste over alle kunder med en sammentælling af ordrer kan man kombinere en outer join med COUNT og GROUP BY.

```
SELECT tblKunder.fldEfternavn, COUNT(tblOrdrer.fldID) as 'Antal ordrer' FROM tblkunder LEFT
OUTER JOIN tblordrer
ON tblkunder.fldID = tblordrer.fldKundeID GROUP BY tblkunder.fldEfternavn ORDER BY
COUNT(tblordrer.fldID) ASC
```

Alle kunder, ordrer eller ej	
Efternavn	**Antal ordrer**
Johnsen	0
Mortensen	0
Nielsen	0
Poulsen	1
Hansen	2
Jensen	2

Hvis man ikke vil have kunder med uden ordre skal disse sorters fra og SQL-statementet kommer til at se ud som nedenstående

SELECT tblKunder.fldEfternavn, COUNT(tblOrdrer.fldID) AS 'Antal ordrer' FROM tblkunder LEFT OUTER JOIN tblordrer ON tblkunder.fldID = tblordrer.fldKundeID WHERE tblordrer.fldID >0 GROUP BY tblkunder.fldEfternavn ORDER BY COUNT(tblordrer.fldID) ASC

hvilket også kan skrives som

SELECT tblKunder.fldEfternavn, COUNT(tblOrdrer.fldID) AS 'Antal ordrer' FROM tblkunder LEFT OUTER JOIN tblordrer ON tblkunder.fldID = tblordrer.fldKundeID WHERE tblordrer.fldID IS NOT NULL GROUP BY tblkunder.fldEfternavn ORDER BY COUNT(tblordrer.fldID) ASC

Kunder, der har afgivet ordrer	
Efternavn	Antal ordrer
Poulsen	1
Hansen	2
Jensen	2

Som det kan ses er resultatet en liste med kundernes efternavn samt det antal ordrer, som den enkelte har afgivet.

SUB-SELECT (En SELECT i en SELECT)

En SUB-SELECT bruges, når man stiller 2 krav /udtræk til et resultatsæt.
Hvis man f. eks. skal finde alle de varer, hvor prisen er højere end gennemsnitsprisen for alle varer.
Dette vil kræve to SELECT-statements.

1. Hvor man finder gennemsnitsprisen for alle varer
   SELECT AVG(fldPris) FROM tblvarer (Svaret er her 5,35)
2. Hvor man finder alle varer, hvis pris er højere end gennemsnittet
   SELECT fldVarenavn, fldPris FROM tblvarer WHERE fldpris > 5.35 (Svaret er meloner til 19 kr.)

Disse to SQL-statements samles

SELECT fldVarenavn, fldPris FROM tblvarer
WHERE fldpris > (SELECT AVG(fldPris) FROM tblvarer)

Svaret er heldigvis også her Meloner til 19 kr.

UNION

UNION bruges, når man skal samle to tabeller.
De to tabeller skal have samme kolonnestruktur for at man kan bruge UNION
For at illustrere dette oprettes en tabel med 2 gamle kunder

Med SQL-statementet

```
SELECT fldFornavn, fldEfternavn FROM tblgamlekunder
UNION
SELECT fldFornavn, fldEfternavn FROM tblkunder
```

får man den viste tabel, der først viser de to kunder fra tabellen med gamle kunder og derefter de 10 nye kunder.

Margrete	Svendsen
Henrik	Jørgensen
Hans	Hansen
Jonna	Jensen
Kirsten	Nielsen
Jacob	Poulsen
Jacob	Nielsen
Karina	Poulsen
Jan	Poulsen
Marina	Johnsen
Carsten	Nielsen
Karsten	Mortensen

HAVING

Man kan også lave klausuler i forbindelse med grupperede resultatsæt. Man kan her ikke bruge WHERE, men skal bruge HAVING.

Vi lavede tidligere et udtræk med GROUP BY, der viste, hvor mange kunder der boede inden for de forskellige postnumre. Hvis vi nu kun er interesseret i at se postnumre indenfor hvilke der bor mere end 2 kunder, skal vi have en HAVING-klausul på udtrækket.

```
SELECT fldPostnr, COUNT(*) AS 'Antal kunder' FROM tblkunder
 GROUP BY fldPostnr HAVING COUNT(*) > 2
```

Dette giver følgende resultat

Postnummer	Antal kunder bosiddende i valgte postnummer
8400	3
8500	3

Ved at bruge HAVING COUNT (*) > 2 bestemmer vi, at SQL-statementet kun må returnere de grupper, som opfylder vores stillede betingelse, nemlig at der skulle bo flere end 2 kunder inden for de forskellige postnumre.

Med HAVING kan man også finde de kunder, der har afgivet mere end 1 ordre.

```
SELECT fldKundeID, COUNT(*) AS 'Antal ordrer' FROM tblordrer
 GROUP BY fldKundeID HAVING COUNT(*) > 1
```

Resultatet af SQL-statementet ses her

KundeID	Antal ordrer
1	2
2	2

Opgave 3

Find de kunder, der har købt æbler (Skal vises med fornavn, varenavn, antal, pris og SUM for den enkelte vare)

Find de kunder, der har købt enten æbler eller appelsiner (Skal vises med fornavn, varenavn, antal, pris og sum for den enkelte vare)

Find kunderne Hansen og Jensen vist med fornavn, efternavn, som har afgivet en ordre

Vis fornavn, efternavn, pris, varenavn på de kunder, som har afgivet en ordre

Vis lagerbeholdningen efter kundernes køb (Varenavn, oprindelig lagerbeholdning, nuværende lagerbeholdning)

Find de varer, der endnu ikke er blevet købt

List kunderne efter hvor meget de har købt

Find den kunde, der har brugt flest penge

List kunderne efter deres samlede køb af enheder. Den med flest enheder skal stå først (fornavn, efternavn, antal)

Find den kunde, der har købt flest enheder

Opgør salget på basis af postnumre

Ændring af data

Hidtil har vi beskæftiget os med udtræk af data fra tabeller. Man kan selvfølgelig også tilføje nye data til en tabel. Rette i eksisterende data i en tabel og slette fra en tabel.

INSERT indsætter nye data i en tabel
UPDATE opdaterer/ændrer data i eksisterende rækker
DELETE sletter

INSERT

INSERT into tblNavn Values(værdi til kolonne 1, værdi til kolonne 2, værdi til kolonne 3 osv.)

Vi vil indsætte en ny vare i vores varetabel, nemlig en ananas. Dette kan gøres med følgende sætning

INSERT INTO tblVarer Values(7, 'Ananas', 24.95, 250)

Denne syntaks kan bruges, når man kender rækkefølgen på kolonnerne. Det er dog mere sikkert at bruge nedenstående syntaks

INSERT INTO tblVarer (fldVarenavn, fldPris, fldAntal) VALUES('Ananas', 24.95, 250)

Man skal her være opmærksom på at de datatyper, som man vil indsætte, passer til de datatyper, der er i tabellen for de på gældende kolonner. Er kolonne 1 et autonummereringsfelt indsættes dette automatisk af SQL og skal derfor ikke stå i Values-listen.

UPDATE

UPDATE kan opdatere såvel enkeltrækker som hele tabeller. Skal man opdatere enkeltrækker, SKAL man HUSKE EN WHERE-KLAUSUL. Man kan også opdatere et enkelt felt i en række eller flere felter i en række.

Glemmer man WHERE-klausulen får UPDATE indflydelse på hele tabellen og man kan IKKE fortryde, når først SQL-sætningen er afviklet.

Selve syntaksen for at opdatere er:

UPDATE tblNavn SET kolonnenavn1 = værdi

Her får alle rækkers kolonnenavn1 den samme værdi. Oftest når man skal opdatere er det kun en enkelt række, man skal opdatere i. Derfor sætter man en betingelse på.

UPDATE tblNavn SET fldKolonnenavn1 = værdi1, fldKolonnenavn2 = værdi2, fldKolonnenavn3 = værdi3, osv WHERE betingelse

Vi opretter en ny vare med INSERT, nemlig en Kiwi, som har en stavefejl, hvor prisen er forkert og antallet mangler (*INSERT INTO tblvarer(fldVarenavn, fldPris) VALUES('Kivi', 4.5))*

Med en UPDATE vil vi rette op på det

UPDATE tblVarer SET fldVarenavn ='Kiwi', fldPris= 4.75, fldAntal=500 WHERE fldID = 7

UPDATE kan nulstille et felts indhold ved at sætte værdien til NULL, hvis det pågældende felt tillader NULL-værdier

UPDATE tblVarer SET fldVarenavn ='Kiwi', fldPris= 4.75, fldAntal=NULL WHERE fldID = 7

Antal er nu sat til NULL. Ret det tilbage til det oprindelige antal inden vi går videre.

DELETE

Med DELETE kan man slette rækker i en tabel.
Man kan slette en række, der opfylder den stillede betingelse.
Man kan slette flere rækker, der alle opfylder den stillede betingelse.
Man kan slette alle rækker.

```
DELETE FROM tblNavn
```

Her sletter man alle rækker i tabellen. Det er dog yderst sjældent, at man ønsker at gøre dette. For det meste vil man kun slette en enkelt række, så derfor skal der en WHERE-klausul på.

Opret en ny tabel, der skal hedde tblTilSlet. Den skal indeholde følgende kolonner; fldID, fldFornavn og fldEfternavn.

```
DELETE FROM tblNavn WHERE betingelse
```

```
DELETE FROM tblTilSlet WHERE fldID = 7
```

Dette SQL-udsagn sletter alle data på kunden med ID-nummer 7

Hvis man vil slette flere rækker ad gangen kan man kombinere DELETE med LIKE

```
DELETE FROM tblTilSlet WHERE fldFornavn LIKE '%ab%'
```

Opgave 4 (INSERT, UPDATE, DELETE)

Varelageret skal udvides med følgende varer:

Varenavn	Pris	Antal
Abrikos	1,00	200
Fersken	4,25	225
Nektarin	4,50	250
Mango	5,00	100

Prisen på Kiwi rettes til 3,25

Antallet af æbler rettes til 1500

Antallet af ananas rettes til 230 og prisen rettes til 21,5

Mango sælger ikke så godt, så den skal slettes igen

Der kommer også flere kunder

Opret derfor følgende kunder

Janni Andersen, Strædet 76, 8000 Århus C, 86 55 55 55, janni@andersen.dk

Janni køber 14 æbler, 23 nektariner

Svend Jørgensen, Nygade 112, 8210 Århus N, 86 44 44 44

Svend køber 4 ananas, 3 meloner, 10 kiwi

Karsten Mortensen har fået en email-adresse: karsten@mortensen.dk

Marina Johnsen har fået nyt telefonnummer: 86 66 66 66

Hans Hansen er flyttet til Bygaden 16 i Hornslet

Jakob Poulsen vil ikke være kunde længere

Opgave 5

Udvid databasen dbFrugt med endnu en tabel. Tabellen skal hedde tblSort.

Den skal indeholde følgende kolonner

Feltnavn	Datatype
fldID	Integer (2), Primary key, NOT NULL, autonummerering
fldGruppe	Varchar (20)

Data skal være:

fldID	fldGruppe
1	frugt
2	bær
3	grønsager

Opret endnu en kolonne i tblVarer, som skal hedde fldGruppeID med datatypen Integer (3)

fldID	fldVarenavn	fldAntal	fldPris	fldGruppeID
1	æbler	1500	1,25	1
2	pærer	2000	2,0	1
3	appelsiner	3000	3,0	1
4	meloner	500	19,0	1
5	bananer	1000	1,5	1
6	Ananas	230	21,5	1
7	Kiwi	750	3,25	1
8	abrikos	200	1,0	1
9	fersken	225	4,25	1
10	nektarin	250	4,5	1
11	ribs	250	4,25	2
12	jordbær	250	12,5	2
13	porrer	500	9,5	3
14	selleri	750	12,75	3

Indsæt nogle varer af de nye typer og sæt typebetegnelse på de første frugter.

Opret også nogle flere ordrer.

fldID	fldVareID	fldKundeID	fldKoebtAntal
1	1	1	10
2	2	1	5
3	1	2	10
4	3	2	10
5	5	6	15
6	1	11	14
7	14	12	16
8	8	12	20
9	12	12	25

Lav nu følgende udtræk:
Vis varenavn og gruppe for alle varer

Vis varenavn og gruppe for alle bær. Resultatet skal vises med varenavn og gruppe

Tæl hvor mange forskellige frugter, der er. Resultatet skal vises som Antal og Gruppe

Find de forskellige grupper, der er i varetabellen

Tæl, hvor mange forskellige grupper, der er i varetabellen

Tæl, hvor mange enheder, der er indenfor hver gruppe i varetabellen

Tæl, hvor mange enheder, der er indenfor hver gruppe i varetabellen. Resultatet skal vises med gruppenavn og antal

Vis alle ordrer med varenavn, gruppenavn, fornavn, efternavn, købssum for de enkelte varer

Vis hvor meget, der er købt for af de enkelte varer

Vis hvor meget, der er købt for indenfor de enkelte grupper

Vis hvor meget den enkelte kunde har købt for

Opgave 6

Opret en ny database. Kald den dbBiler

Opret følgende tabeller:

- tblMaerke
- tblFarve
- tblBiler
- tblHK

tblMaerke	tblFarve	tblHK
Opel	Rød	75
Peugeot	Blå	90
VW	Hvid	100
Opel	Sort	110
Audi	Sølv	120
Skoda	Grøn	130
		140

tblBiler :

fldMaerke	fldFarve	fldModel	fldHK	fldKilometre
Opel	Rød	Astra	100	100000
Peugeot	Sort	308	140	90000
Peugeot	Sølv	307	110	80000
Peugeot	Blå	307	110	110000
Audi	Sølv	A4	130	60000
VW	Rød	Golf	100	120000
VW	Sort	Golf TDI	140	100000

Sørg for at biltabellen bliver normaliseret.

Lav derefter følgende udtræk, hvor resultatet skal vises med mærke, model, farve, HK og kørte kilometre for hvert enkelt udtræk.

Find alle de biler, der har kørt over 75.000 kilometer.

Find alle biler, der har 110 HK eller mindre under motorhjelmen

Find alle blå peugeoter model 307

Find alle de røde biler

Find alle de biler, der er sorte, har kørt 100.000 km eller mere og har 120 HK eller mere

Indsæt en ny bil

Skoda, Felicia, 90HK, 75.000km, blå

Indsæt endnu en bil

VW, TDI, 75HK, 65.000 km, sort

Den bil, du lige har oprettet skal være hvid og den har i stedet kørt 30.000 km

Opret en ny bil

Skoda, Felicia, hvid, 40.000 km, 75 HK

Slet denne bil igen

Vis alle biler

Hvor mange biler der er i alt?

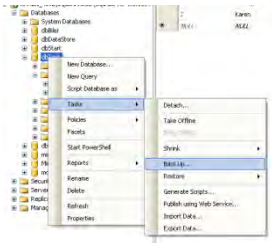
Hvor mange biler er der inden for hvert mærke?

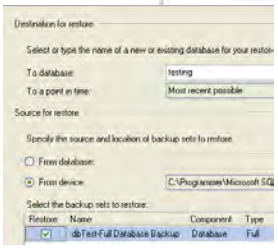
Har alle farver været i brug?

Backup af database

Man skal huske at tage backup af ens databaser. For programmet eller databasen kan gå ned en gang imellem. Ved at tage en backup af sin database giver man også mulighed for at databasen kan importeres af andre.

Databasen findes jo ikke fysisk, når man bruger MySQL eller MSSQL. Så man gemmer faktisk sin database som SQL.



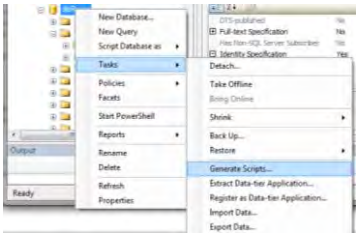


MSSQL/MSSQLEXPRESS

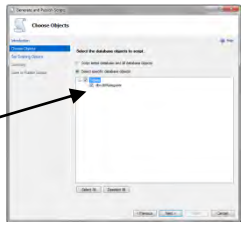
Arbejder man med MSSQL og skal lave backup, så højreklikker man på den database, man vil lave backup af. Her vælges Task og herefter Back Up. Når man har valgt dette får man et nyt vindue. Her står navnet på den database, man ønsker backup af. Den kan navngives med et andet navn. Under destination peger MSSQL allerede på en placering for backup-filen men man kan sagtens gemme den et andet sted. Så klikker man på Add og finder den placering, hvor man vil gemme sin backup. Når databasebackup'en er gennemført, får man et svar om at det er sket.

Når man så skal gendanne sin database, højreklikker man på Databases. Vælger Restore. Her kan man navngive sin database. Vælg From device og find det sted, hvor backup-filen er gemt. Vælg den. Husk at sætte flueben under Restore. Klik herefter OK og databasen blever lavet.

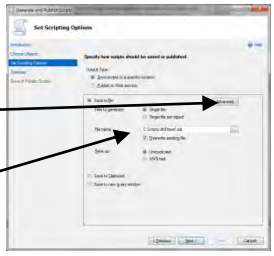
Man behøver ikke lave en backup af sine daabaser, Man kan også scripte dem. Dette gøres ved at højreklikke på databasen. Vælge menupunktet Tasks og herefter Generate scripts. Man kan her vælge at lave et script, der kan generere hele databasen eller man kan lave et script, der kan generere tabellerne med data i den pågøældende database. Her vil vi generere en script, der kan gendanne vores tabeller med data.



Når man har valgt Generate Scripts, kommer denne dialogboks frem.
Klik på Next



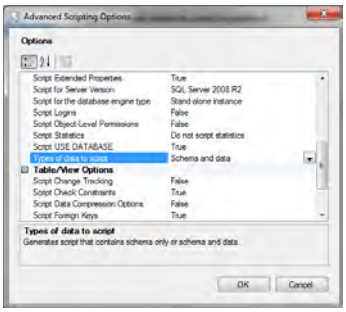
På det næste dialogboks skal man vælge punktet Select specific database objects og huske at sætte flueben i Tables. Det er en god ide, at tjekke, at der er flueben i alle de tabeller, man gerne vil have scriptet.
Klik på Next



I denne diaglogboks skal man angive, hvor man vil gemme sit script og hvad det skal hedde. Det kan være en god ide, at give scriptet databasens navn og have en speciel mappe til scripts

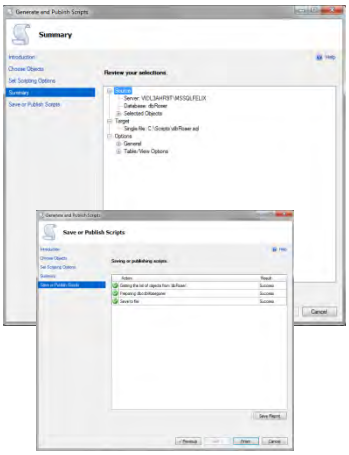
Klik på knappen Advanced

I den næstedialogboks skal man finde punktet *Types of data to script* og sætte værdien til *Schema and data*



Klik på Ok. Man kommer så tilbage til den foregående dialogboks og her klikker man nu Next

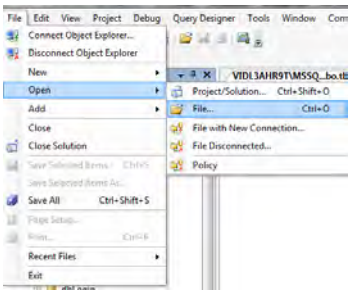
Næste dialogboks er et tjek, så man se, hvad, der bliver gemt og hvorhenne. Har man valgt forkert, kan man komme tilbage ved at klikke på knappen Previous.



Er betingelserne korrekte klikker man på Next og programmet genererer en sql-fil, som indeholder struktur og data for de valgte tabeller.

Klik derefter på finish

Når man så skal køre scriptet for at få gendannet sine tabeller skal man indlæse det i SQL Management Studio. Dette gør man ved at vælge menupunktet File → Open → File. Finde sin fil og klikke på Open. Scriptet bliv er så indlæst og vil se ud som dette

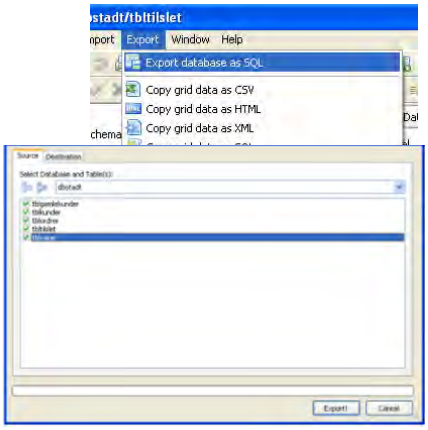


Man skal her være opmærksom på, at det er den rigtige database, der henvises til. Er det ikke det, skal man rette det. Klik på Execute og tabellerne dannes i den valgte database.

```
1  USE [dbRoser]
2  GO
3  /****** Object:  Table [dbo].[tblKategorier]    Script Date: 08/07/2012 12:08:12 ******/
4  SET ANSI_NULLS ON
5  GO
6  SET QUOTED_IDENTIFIER ON
7  GO
8  SET ANSI_PADDING ON
9  GO
10 CREATE TABLE [dbo].[tblKategorier](
11     [fldID] [int] IDENTITY(1,1) NOT NULL,
12     [fldKategori] [varchar](20) NULL,
13  CONSTRAINT [PK_tblKategorier] PRIMARY KEY CLUSTERED
14 (
15     [fldID] ASC
16 )WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS  = ON,
   ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY]
17 ) ON [PRIMARY]
18 GO
19 SET ANSI_PADDING OFF
20 GO
21 SET IDENTITY_INSERT [dbo].[tblKategorier] ON
22 INSERT [dbo].[tblKategorier] ([fldID], [fldKategori]) VALUES (1, N'Engelske roser')
23 INSERT [dbo].[tblKategorier] ([fldID], [fldKategori]) VALUES (2, N'Klatreroser')
24 INSERT [dbo].[tblKategorier] ([fldID], [fldKategori]) VALUES (3, N'Slyngroser')
25 SET IDENTITY_INSERT [dbo].[tblKategorier] OFF
26
```

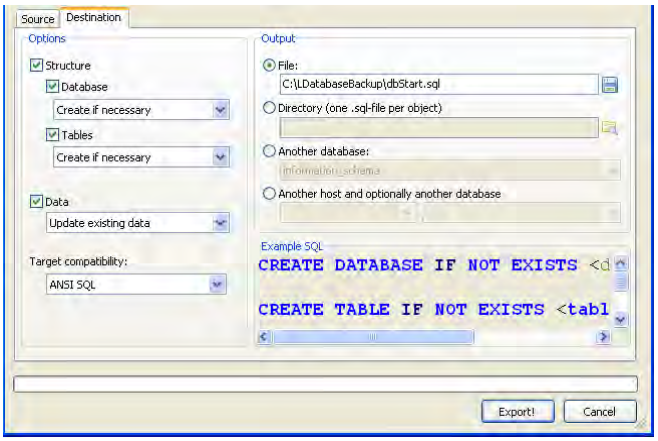
MySQL

Når man skal tage backup vælger man i Heidi menupunktet Export og herunder Export database as SQL.



Under fanebladet Source skal man huske at sætte flueben ud for tabelnavnene. Eller kommer de ikke med.

Under fanebladet Destination skal man angive, hvor man vil gemme backup-filen. Her er det på C-drevet i en mappe, der hedder LDatabaseBackup. Filen kommer til at hedde dbFrugt.sql.



Klik derefter på Export!

Det er nu muligt at importere databasen og arbejde videre med den eller se dens opbygning.

Nyttige links

MySQL's hjemmeside www.mySQL.com (her findes de forskellige downloads til MySQL)
Heidis hjemmeside www.HeidiSQL.com (download af Heidi)

SQL-syntaks: www.w3schools.org (SQL-syntaks til de forskellige databaser)
MSSQL-syntaks: http://msdn.microsoft.com/en-us/library/aa213227(SQL.80).aspx

Forbindelse til database fra ASP.NET ved hjælp af C# (MSSQL)

Vi er nu nået så langt, at vi skal til at forbinde vores .NET site med vores database.

Man kan lave forbindelsen på 3 måder, som alle vil blive vist i nedenstående eksempler.

Man kan have sin forbindelse stående i web.config-filen. Man kan have den i sin dataAccess-fil eller man kan have den i sine factories.

Eksempel på forbindelse via web.config-filen til C#

Som udgangspunkt for vores tre små eksempler vil vi bruge den database, som vi har, nemlig dbFrugt.

Vi laver først et eksempel, hvor forbindelsen hentes via web.config-filen.

Man kan i sine sites sagtens have forbindelser til flere forskellige databaser i web.config-filen.

Opret et nyt website.

Kald det startMedWebconfigCSMSSQL.

I webconfig-filen under <connectionStrings> skrives følgende

```
<connectionStrings>
                        <add name="NavnPaaForbindelse"
connectionString="server=Servernavn;database=databasenavn;uid=brugeridentifikati
on;pwd=password"/>
            </connectionStrings>
```

Opret dernæst dbDataAccess.cs i App_Code-mappen

dbDataAccess.cs

```
using System.Data;
using System.Configuration;
using System.Data.SqlClient;

public class dbDataAccess
{
    string strCon = ConfigurationManager.ConnectionStrings["NavnPaaForbindelse
"].ConnectionString;
    //giver adgang til de funktioner, der ligger i dbDataAccess.cs
    public DataTable GetData(SqlCommand CMD)
           {
        //Ny forbindelse fra Connectionstring'en
        SqlConnection objConn = new SqlConnection(strCon);

        //Ny SqlDataAdapter
        SqlDataAdapter objDA = new SqlDataAdapter();

         //Nyt Dataset
        DataSet objDS = new DataSet();

        //Angivelse af SqlCommando'ens forbindelse
        CMD.Connection = objConn;
```

```
        //Angivelse af DataAdapterens Select-kommando (da getData kun bruges ved
SELECT-statemants)
        objDA.SelectCommand = CMD;
        objDA.Fill(objDS);

        return objDS.Tables[0];


           }

    public void ModifyData(SqlCommand CMD)
    {
        //Ny forbindelse fra Connectionstring'en
        SqlConnection objConn = new SqlConnection(strCon);

        //Kommandoens forbindelse
        CMD.Connection = objConn;

        //Åbn forbindelsen
        objConn.Open();

        //Udfør SQL-kommandoen (en UPDATE/INSERT/DELETE) der ikke giver et
resultatsæt men derimod et tal, nemlig antallet af påvirkede rækker
        CMD.ExecuteNonQuery();
        objConn.Close();

        }
}
```

Klassen her indeholder både en metode, der kan give adgang til at trække data ud og en metode, der giver adgang til at ændre data i databasen.

Opret filen udtraekFac.cs i App_Code-mappen

Lav her en funktion, der trækker alle varenavne ud

```
using System.Data;
using System.Data.SqlClient;

public class udtraekFac
{
    //giver adgang til de funktioner, der ligger i dbDataAccess.cs
    dbDataAccess objConn = new dbDataAccess();

    //metode, der trækker alt ud fra tabellen tblVarer i databasen dbFrugt
    public DataTable getAlleVarer()
    {
        //Laver den sql-streng, der skal bruges
        string strSQL = "Select fldID, fldVarenavn, fldPris, fldAntal From
tblvarer order by fldID desc";

        //bruger CommandObjektet til at gemme resultatsættet i
        SqlCommand objCMD = new SqlCommand(strSQL);

        return objConn.GetData(objCMD);
```

```
    }

}
```

På default.aspx trækkes to literals ind.

Den første skal være til beskrivende tekst, den anden til at vise vores data i.

```
    <form id="form1" runat="server">
    <div>
        <asp:Literal ID="litOverskrift" runat="server"></asp:Literal><br /><br />
        <asp:Literal ID="litVis" runat="server"></asp:Literal>
    </div>
    </form>
```

Default.aspx.cs

```
using System;
using System.Data;

public partial class _Default : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {
        if (!IsPostBack)
        {
            litOverskrift.Text = "Her vises alle varenavne i databasen dbFrugt<br />";
            litOverskrift.Text += "Her ligger forbindelsen til databasen i web.config-filen";

            udtraekFac objUdtraek = new udtraekFac();

            foreach (DataRow row in objUdtraek.getAlleVarer().Rows)
            {
                litVis.Text += "Varenavn: " + row["fldVarenavn"] + "<br />Pris: " + row["fldPris"] + " kr.<hr /><br />";
            }

        }
    }
}
```

Eksempel på forbindelse via dbDataAccess.cs til C#

Opret et nyt website. Kald det startMeddbDataAccessCSMSSQL

I mappen App_Code oprettes filen dbDataAccess.cs

```
using System.Data.SqlClient;
using System.Data;

public class dbDataAccess
```

```
{
        string strCon =
"server=Servernavn;database=databasenavn;uid=brugeridentifikation;pwd=password";
    public DataTable GetData(SqlCommand CMD)
           {
        //Ny forbindelse fra Connectionstring'en
        SqlConnection objConn = new SqlConnection(strCon);

        //Ny SqlDataAdapter
        SqlDataAdapter objDA = new SqlDataAdapter();

         //Nyt Dataset
        DataSet objDS = new DataSet();

        //Angivelse af SqlCommando'ens forbindelse
        CMD.Connection = objConn;

        //Angivelse af DataAdapterens Select-kommando (da getData kun bruges ved
SELECT-statemants)
        objDA.SelectCommand = CMD;
        objDA.Fill(objDS);

        return objDS.Tables[0];


           }

    public void ModifyData(SqlCommand CMD)
    {
        //Ny forbindelse fra Connectionstring'en
        SqlConnection objConn = new SqlConnection(strCon);

        //Kommandoens forbindelse
        CMD.Connection = objConn;

        //Åbn forbindelsen
        objConn.Open();

        //Udfør SQL-kommandoen (en UPDATE/INSERT/DELETE) der ikke giver et
resultatsæt men derimod et tal, nemlig antallet af påvirkede rækker
        CMD.ExecuteNonQuery();
        objConn.Close();

        }
}
```

Opret også filen udtraekFac.cs i mappen App_Code

```
using System.Data;
using System.Data.SqlClient;

public class udtraekFac
{
    //giver adgang til de funktioner, der ligger i dbDataAccess.cs
    dbDataAccess objConn = new dbDataAccess();

    //metode, der trækker alt ud fra tabellen tblVarer i databasen dbFrugt
    public DataTable getAlleVarer()
```

```
    {
        //Laver den sql-streng, der skal bruges
     string strSQL = "Select fldID, fldVarenavn, fldPris, fldAntal From tblvarer
order by fldID desc";

        //bruger CommandObjektet til at gemme resultatsættet i
        SqlCommand objCMD = new SqlCommand(strSQL);

        return objConn.GetData(objCMD);
    }

}
```

På default.aspx trækkes to literals ind.

Den første skal være til beskrivende tekst, den anden til at vise vores data i.

```
    <form id="form1" runat="server">
    <div>
        <asp:Literal ID="litOverskrift" runat="server"></asp:Literal><br /><br />
        <asp:Literal ID="litVis" runat="server"></asp:Literal>
    </div>
    </form>
```

Default.aspx.cs

```
using System.Data;

public partial class _Default : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {
        if (!IsPostBack)
        {
            litOverskrift.Text = "Her vises alle varenavne i databasen
dbFrugt<br />";
            litOverskrift.Text += "Her ligger forbindelsen til databasen i
dbDataAccess-filen";

            udtraekFac objUdtraek = new udtraekFac();

            foreach (DataRow row in objUdtraek.getAlleVarer().Rows)
            {
                litVis.Text += "Varenavn: " + row["fldVarenavn"] + "<br />Pris:
" + row["fldPris"] + " kr.<hr /><br />";
            }

        }
    }
}
```

Eksempel på forbindelse via factories til C#

Opret et nyt website. Kald det startMedFactoriesCSMSSQL

Opret filen dbDataAccess.cs i mappen App_Code

```
using System.Data;
using System.Data.SqlClient;

public class dbDataAccess
{
    public DataTable GetData(SqlCommand CMD, SqlConnection CON)
    {

        //Ny SqlDataAdapter
        SqlDataAdapter objDA = new SqlDataAdapter();

        //Nyt Dataset
        DataSet objDS = new DataSet();

        //Angivelse af SqlCommando'ens forbindelse
        CMD.Connection = CON;

        //Angivelse af DataAdapterens Select-kommando (da getData kun bruges ved
SELECT-statemants)
        objDA.SelectCommand = CMD;
        objDA.Fill(objDS);

        return objDS.Tables[0];
    }

    public void ModifyData(SqlCommand CMD, SqlConnection CON)
    {
        CMD.Connection = CON;
        CON.Open();
        CMD.ExecuteNonQuery();
        CON.Close();
    }
}
```

Klassen her indeholder både en metode, der kan give adgang til at trække data ud og en metode, der giver adgang til at ændre data i databasen.

Opret også filen udtraekFac.cs i mappen App_Code

```
using System.Data;
using System.Data.SqlClient;

public class udtraekFac
{
    //giver adgang til de funktioner, der ligger i dbDataAccess.vb
     //forbindelse til database
    string strCon =
"server=Servernavn;database=databasenavn;uid=brugeridentifikation;pwd=password";

    dbDataAccess objConn = new dbDataAccess();
```

```
    //metode, der trækker alt ud fra tabellen tblVarer i databasen dbFrugt
    public DataTable getAlleVarer()
    {
        //Laver den sql-streng, der skal bruges
        string strSQL = "Select fldID, fldVarenavn, fldPris, fldAntal From
tblvarer order by fldID desc";

        //bruger CommandObjektet til at gemme resultatsættet i
        SqlCommand objCMD = new SqlCommand(strSQL);
        SqlConnection _Con = new SqlConnection(strCon);
        return objConn.GetData(objCMD, _Con);
    }

}
```

På default.aspx trækkes to literals ind.

Den første skal være til beskrivende tekst, den anden til at vise vores data i.

```
    <form id="form1" runat="server">
    <div>
        <asp:Literal ID="litOverskrift" runat="server"></asp:Literal><br /><br />
        <asp:Literal ID="litVis" runat="server"></asp:Literal>
    </div>
    </form>
```

Default.aspx.cs

```
using System.Data;
using System.Linq;

public partial class _Default : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {
        if (!IsPostBack)
        {
            litOverskrift.Text = "Her vises alle varenavne i databasen
dbFrugt<br />";
            litOverskrift.Text += "Her ligger forbindelsen til databasen i de
enkelte factories";

            udtraekFac objUdtraek = new udtraekFac();

            foreach (DataRow row in objUdtraek.getAlleVarer().Rows)
            {
                litVis.Text += "Varenavn: " + row["fldVarenavn"] + "<br />Pris:
" + row["fldPris"] + " kr.<hr /><br />";
            }
        }
    }
}
```

Man kan selvfølgelig sagtens samle de forskellige forbindelser i en fil (til C#)

dbDataAccessSamlet.cs

```
using System.Data;
using System.Configuration;
using System.Data.SqlClient;

public class dbConnSamlet
{
//------------------------------------------------------
    string strCon =
ConfigurationManager.ConnectionStrings["NavnPaaForbindelse"].ConnectionString;
    //giver adgang til de metoder, der ligger i dbDataAccess.cs

    public DataTable GetDataWebConfig(SqlCommand CMD)
    {
        //Ny forbindelse fra Connectionstring'en
        SqlConnection objConn = new SqlConnection(strCon);

        //Ny MySqlDataAdapter
        SqlDataAdapter objDA = new SqlDataAdapter();

        //Nyt Dataset
        DataSet objDS = new DataSet();

        //Angivelse af SqlCommando'ens forbindelse
        CMD.Connection = objConn;

        //Angivelse af DataAdapterens Select-kommando (da getData kun bruges ved
SELECT-statemants)
        objDA.SelectCommand = CMD;
        objDA.Fill(objDS);

        return objDS.Tables[0];
    }

    public void ModifyDataWebConfig(SqlCommand CMD)
    {
        //Ny forbindelse fra Connectionstring'en
        SqlConnection objConn = new SqlConnection(strCon);

        //Kommandoens forbindelse
        CMD.Connection = objConn;

        //Åbn forbindelsen
        objConn.Open();

        //Udfør SQL-kommandoen (en UPDATE/INSERT/DELETE) der ikke giver et
resultatsæt men derimod et tal, nemlig antallet af påvirkede rækker
        CMD.ExecuteNonQuery();
        objConn.Close();

    }


    // ------------------------------------------------------------------
```

```
    string strCon =
"server=Servernavn;database=databasenavn;uid=brugeridentifikation;pwd=password";
    public DataTable GetDatadbDataAccess(SqlCommand CMD)
            {
        //Ny forbindelse fra Connectionstring'en
        SqlConnection objConn = new SqlConnection(strCon);

        //Ny SqlDataAdapter
        SqlDataAdapter objDA = new SqlDataAdapter();

         //Nyt Dataset
        DataSet objDS = new DataSet();

        //Angivelse af SqlCommando'ens forbindelse
        CMD.Connection = objConn;

        //Angivelse af DataAdapterens Select-kommando (da getData kun bruges ved
SELECT-statemants)
        objDA.SelectCommand = CMD;
        objDA.Fill(objDS);

        return objDS.Tables[0];


           }

    public void ModifyDatadbAccess(SqlCommand CMD)
    {
        //Ny forbindelse fra Connectionstring'en
        SqlConnection objConn = new SqlConnection(strCon);

        //Kommandoens forbindelse
        CMD.Connection = objConn;

        //Åbn forbindelsen
        objConn.Open();

        //Udfør SQL-kommandoen (en UPDATE/INSERT/DELETE) der ikke giver et
resultatsæt men derimod et tal, nemlig antallet af påvirkede rækker
        CMD.ExecuteNonQuery();
        objConn.Close();

        }
    // ------------------------------------------------------------------

    public DataTable GetDataFactories(SqlCommand CMD, SqlConnection CON)
    {

        //Ny SqlDataAdapter
        SqlDataAdapter objDA = new SqlDataAdapter();

        //Nyt Dataset
        DataSet objDS = new DataSet();

        //Angivelse af SqlCommando'ens forbindelse
        CMD.Connection = CON;

        //Angivelse af DataAdapterens Select-kommando (da getData kun bruges ved
SELECT-statemants)
```

```
    objDA.SelectCommand = CMD;
    objDA.Fill(objDS);

    return objDS.Tables[0];
}

public void ModifyDataFactories(SqlCommand CMD, SqlConnection CON)
{
    CMD.Connection = CON;
    CON.Open();
    CMD.ExecuteNonQuery();
    CON.Close();
}

// ------------------------------------------------------------------
       }
```

Forbindelse til database fra ASP.NET ved hjælp af C# (MySQL)

Vi er nu nået så langt, at vi skal til at forbinde vores .NET site med vores database.

Man kan lave forbindelsen på 3 måder, som alle vil blive vist i nedenstående eksempler.

Man kan have sin forbindelse stående i web.config-filen. Man kan have den i sin dataAccess-fil eller man kan have den i sine factories.

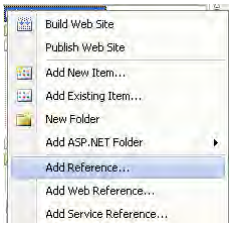
Eksempel på forbindelse via web.config-filen til C#

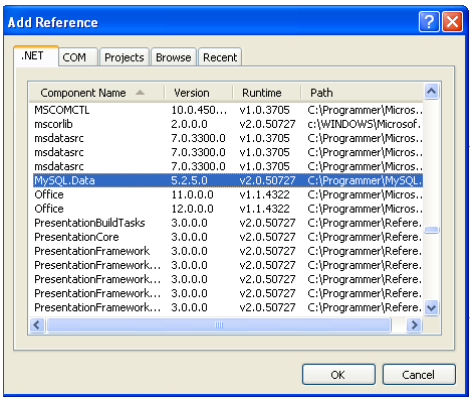
Som udgangspunkt for vores tre små eksempler vil vi bruge den database, som vi har, nemlig dbFrugt.

Vi laver først et eksempel, hvor forbindelsen hentes via web.config-filen.

Opret et nyt website.

Kald det startMedWebconfigCS. Husk at tilføje en reference til MySQL. Ellers får man ikke adgang til de namespaces, man skal bruge, når ens database er oprettet med MySQL. En reference tilføjes ved at højreklikke på roden i Solution Explorer, vælge punktet Add Reference. Her vælges .NET-fanebladet og MySQL.Data findes.





Den vælges og der klikkes OK.

I webconfig-filen under <connectionStrings> skrives

```
        <connectionStrings>
                <add name="NavnPaaStreng"
connectionString="server=servernavn;database=databasenavn;uid=brugernavn;pwd=pas
sword;"/>
        </connectionStrings>
```

I webconfig-filen kan man godt have flere forbindelser til forskellige databaser. F. eks. både til MySQL og MSSQL

Opret dernæst dbDataAccess.cs i App_Code-mappen

dbDataAccess.cs

```
using System.Data;
using System.Configuration;
using MySql.Data.MySqlClient;

public class dbDataAccess
{
    string strCon =
ConfigurationManager.ConnectionStrings["NavnPaaStreng"].ConnectionString;
    //giver adgang til de funktioner, der ligger i dbDataAccess.cs
    public DataTable GetData(MySqlCommand CMD)
```

```
            {
        //Ny forbindelse fra Connectionstring'en
        MySqlConnection objConn = new MySqlConnection(strCon);

        //Ny MySqlDataAdapter
        MySqlDataAdapter objDA = new MySqlDataAdapter();

         //Nyt Dataset
        DataSet objDS = new DataSet();

        //Angivelse af MySqlCommando'ens forbindelse
        CMD.Connection = objConn;

        //Angivelse af DataAdapterens Select-kommando (da getData kun bruges ved
SELECT-statemants)
        objDA.SelectCommand = CMD;
        objDA.Fill(objDS);

        return objDS.Tables[0];


            }

    public void ModifyData(MySqlCommand CMD)
    {
        //Ny forbindelse fra Connectionstring'en
        MySqlConnection objConn = new MySqlConnection(strCon);

        //Kommandoens forbindelse
        CMD.Connection = objConn;

        //Åbn forbindelsen
        objConn.Open();

        //Udfør SQL-kommandoen (en UPDATE/INSERT/DELETE) der ikke giver et
resultatsæt men derimod et tal, nemlig antallet af påvirkede rækker
        CMD.ExecuteNonQuery();
        objConn.Close();

        }

}
```

Klassen her indeholder både en metode, der kan give adgang til at trække data ud og en metode, der giver adgang til at ændre data i databasen.

Opret filen udtraekFac.cs i App_Code-mappen

Lav her en funktion, der trækker alle varenavne ud

```
using System.Data;
using MySql.Data.MySqlClient;

public class udtraekFac
{
    //giver adgang til de funktioner, der ligger i dbDataAccess.cs
    dbDataAccess objConn = new dbDataAccess ();
```

```
    //Funktion, der trækker alt ud fra tabellen tblVarer i databasen dbFrugt
    public DataTable getAlleVarer()
    {
        //Laver den sql-streng, der skal bruges
        string strSQL = "Select fldID, fldVarenavn, fldPris, fldAntal From
tblvarer order by fldID desc";

        //bruger CommandObjektet til at gemme resultatsættet i
        MySqlCommand objCMD = new MySqlCommand(strSQL);

        return objConn.GetData(objCMD);
    }

}
```

På default.aspx trækkes to literals ind.

Den første skal være til beskrivende tekst, den anden til at vise vores data i.

```
    <form id="form1" runat="server">
    <div>
        <asp:Literal ID="litOverskrift" runat="server"></asp:Literal><br /><br />
        <asp:Literal ID="litVis" runat="server"></asp:Literal>
    </div>
    </form>
```

Default.aspx.cs

```
using System.Data;

public partial class _Default : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {
        if (!IsPostBack)
        {
            litOverskrift.Text = "Her vises alle varenavne i databasen
dbFrugt<br />";
            litOverskrift.Text += "Her ligger forbindelsen til databasen i
webconfig-filen";

            udtraekFac objUdtraek = new udtraekFac();

            foreach (DataRow row in objUdtraek.getAlleVarer().Rows)
            {
                litVis.Text += "Varenavn: " + row["fldVarenavn"] + "<br />Pris:
" + row["fldPris"] + " kr.<hr /><br />";
            }

        }
    }
}
```

Eksempel på forbindelse via dbDataAccess.cs til C#

Opret et nyt website. Kald det startMeddbDataAccessCS

I mappen App_Code oprettes filen dbDataAccess.cs

```
using System.Data;
using MySql.Data.MySqlClient;

public class dbDataAccess
{
    string strCon =
"Server=servernavn;Database=databasenavn;Uid=brugeridentifikation;Pwd=password;"
;

    public DataTable GetData(MySqlCommand CMD)
           {
        //Ny forbindelse fra Connectionstring'en
        MySqlConnection objConn = new MySqlConnection(strCon);

        //Ny MySqlDataAdapter
        MySqlDataAdapter objDA = new MySqlDataAdapter();

         //Nyt Dataset
        DataSet objDS = new DataSet();

        //Angivelse af MySqlCommando'ens forbindelse
        CMD.Connection = objConn;

        //Angivelse af DataAdapterens Select-kommando (da getData kun bruges ved
SELECT-statemants)
        objDA.SelectCommand = CMD;
        objDA.Fill(objDS);

        return objDS.Tables[0];


           }

    public void ModifyData(MySqlCommand CMD)
    {
        //Ny forbindelse fra Connectionstring'en
        MySqlConnection objConn = new MySqlConnection(strCon);

        //Kommandoens forbindelse
        CMD.Connection = objConn;

        //Åbn forbindelsen
        objConn.Open();

        //Udfør SQL-kommandoen (en UPDATE/INSERT/DELETE) der ikke giver et
resultatsæt men derimod et tal, nemlig antallet af påvirkede rækker
        CMD.ExecuteNonQuery();
        objConn.Close();

        }
}
```

Opret også filen udtraekFac.cs i mappen App_Code

```
using System.Data;
using MySql.Data.MySqlClient;

public class udtraekFac
{
    //giver adgang til de metoder, der ligger i dbDataAccess.cs
    dbDataAccess objConn = new dbDataAccess ();

    //metoder, der trækker alt ud fra tabellen tblVarer i databasen dbFrugt
    public DataTable getAlleVarer()
    {
        //Laver den sql-streng, der skal bruges
     string strSQL = "Select fldID, fldVarenavn, fldPris, fldAntal From tblvarer
order by fldID desc";

        //bruger CommandObjektet til at gemme resultatsættet i
        MySqlCommand objCMD = new MySqlCommand(strSQL);

        return objConn.GetData(objCMD);
    }

}
```

På default.aspx trækkes to literals ind.

Den første skal være til beskrivende tekst, den anden til at vise vores data i.

```
    <form id="form1" runat="server">
    <div>
        <asp:Literal ID="litOverskrift" runat="server"></asp:Literal><br /><br />
        <asp:Literal ID="litVis" runat="server"></asp:Literal>
    </div>
    </form>
```

Default.aspx.cs

```
using System.Data;
public partial class _Default : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {
        if (!IsPostBack)
        {
            litOverskrift.Text = "Her vises alle varenavne i databasen
dbFrugt<br />";
            litOverskrift.Text += "Her ligger forbindelsen til databasen i
dbDataAccess-filen";

            udtraekFac objUdtraek = new udtraekFac();

            foreach (DataRow row in objUdtraek.getAlleVarer().Rows)
            {
```

```
                litVis.Text += "Varenavn: " + row["fldVarenavn"] + "<br />Pris:
" + row["fldPris"] + " kr.<hr /><br />";
            }
        }
    }
}
```

Eksempel på forbindelse via factories til C#

Opret et nyt website. Kald det startMedFactoriesCS

Opret filen dbDataAccess.cs i mappen App_Code

```
using System.Data;
using MySql.Data.MySqlClient;

public class dbDataAccess
{
    public DataTable GetData(MySqlCommand CMD, MySqlConnection CON)
    {

        //Ny MySqlDataAdapter
        MySqlDataAdapter objDA = new MySqlDataAdapter();

        //Nyt Dataset
        DataSet objDS = new DataSet();

        //Angivelse af MySqlCommando'ens forbindelse
        CMD.Connection = CON;

        //Angivelse af DataAdapterens Select-kommando (da getData kun bruges ved
SELECT-statemants)
        objDA.SelectCommand = CMD;
        objDA.Fill(objDS);

        return objDS.Tables[0];
    }

    public void ModifyData(MySqlCommand CMD, MySqlConnection CON)
    {
        CMD.Connection = CON;
        CON.Open();
        CMD.ExecuteNonQuery();
        CON.Close();
    }
}
```

Klassen her indeholder både en metode, der kan give adgang til at trække data ud og en metode, der giver adgang til at ændre data i databasen.

Opret også filen udtraekFac.cs i mappen App_Code

```
using System.Data;
using MySql.Data.MySqlClient;

public class udtraekFac
{
```

```
    //giver adgang til de metoder, der ligger i dbDataAccess.cs
     //forbindelse til database
    string strCon =
"Server=servernavn;Database=databasenavn;Uid=brugeridentifikation;Pwd=password;";

    dbDataAccess objConn = new dbDataAccess();

    //metode, der trækker alt ud fra tabellen tblVarer i databasen dbFrugt
    public DataTable getAlleVarer()
    {
        //Laver den sql-streng, der skal bruges
        string strSQL = "Select fldID, fldVarenavn, fldPris, fldAntal From
tblvarer order by fldID desc";

        //bruger CommandObjektet til at gemme resultatsættet i
        MySqlCommand objCMD = new MySqlCommand(strSQL);
        MySqlConnection _Con = new MySqlConnection(strCon);
        return objConn.GetData(objCMD, _Con);
    }

}
```

På default.aspx trækkes to literals ind.

Den første skal være til beskrivende tekst, den anden til at vise vores data i.

```
    <form id="form1" runat="server">
    <div>
        <asp:Literal ID="litOverskrift" runat="server"></asp:Literal><br /><br />
        <asp:Literal ID="litVis" runat="server"></asp:Literal>
    </div>
    </form>
```

Default.aspx.cs

```
using System.Data;
using System.Linq;

public partial class _Default : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {
        if (!IsPostBack)
        {
            litOverskrift.Text = "Her vises alle varenavne i databasen
dbFrugt<br />";
            litOverskrift.Text += "Her ligger forbindelsen til databasen i de
enkelte factories";

            udtraekFac objUdtraek = new udtraekFac();

            foreach (DataRow row in objUdtraek.getAlleVarer().Rows)
            {
                litVis.Text += "Varenavn: " + row["fldVarenavn"] + "<br />Pris:
" + row["fldPris"] + " kr.<hr /><br />";
```

```
            }
        }
    }
}
```

Man kan selvfølgelig sagtens samle de forskellige forbindelser i en fil (til C#)

dbDataAccessSamlet.cs

```
using System.Data;
using System.Configuration;
using MySql.Data.MySqlClient;

public class dbDataAccessSamlet
{
    //-------------------------------------------------------

    string strCon =
ConfigurationManager.ConnectionStrings["NavnPaaForbindelse"].ConnectionString;
    //giver adgang til de funktioner, der ligger i dbConn.vb

    public DataTable GetDataWebConfig(MySqlCommand CMD)
    {
        //Ny forbindelse fra Connectionstring'en
        MySqlConnection objConn = new MySqlConnection(strCon);

        //Ny MySqlDataAdapter
        MySqlDataAdapter objDA = new MySqlDataAdapter();

        //Nyt Dataset
        DataSet objDS = new DataSet();

        //Angivelse af MySqlCommando'ens forbindelse
        CMD.Connection = objConn;

        //Angivelse af DataAdapterens Select-kommando (da getData kun bruges ved
SELECT-statemants)
        objDA.SelectCommand = CMD;
        objDA.Fill(objDS);

        return objDS.Tables[0];


    }

    public void ModifyDataWebConfig(MySqlCommand CMD)
    {
        //Ny forbindelse fra Connectionstring'en
        MySqlConnection objConn = new MySqlConnection(strCon);

        //Kommandoens forbindelse
        CMD.Connection = objConn;

        //Åbn forbindelsen
        objConn.Open();
```

```
        //Udfør SQL-kommandoen (en UPDATE/INSERT/DELETE) der ikke giver et
resultatsæt men derimod et tal, nemlig antallet af påvirkede rækker
        CMD.ExecuteNonQuery();
        objConn.Close();

    }


    // ------------------------------------------------------------------
    string strCon =
"Server=servernavn;Database=databasenavn;Uid=brugeridentification;Pwd=password;";

    public DataTable GetDatadbDataAccess(MySqlCommand CMD)
           {
        //Ny forbindelse fra Connectionstring'en
        MySqlConnection objConn = new MySqlConnection(strCon);

        //Ny MySqlDataAdapter
        MySqlDataAdapter objDA = new MySqlDataAdapter();

         //Nyt Dataset
        DataSet objDS = new DataSet();

        //Angivelse af MySqlCommando'ens forbindelse
        CMD.Connection = objConn;

        //Angivelse af DataAdapterens Select-kommando (da getData kun bruges ved
SELECT-statemants)
        objDA.SelectCommand = CMD;
        objDA.Fill(objDS);

        return objDS.Tables[0];


           }

    public void ModifyDatadbDataAccess(MySqlCommand CMD)
    {
        //Ny forbindelse fra Connectionstring'en
        MySqlConnection objConn = new MySqlConnection(strCon);

        //Kommandoens forbindelse
        CMD.Connection = objConn;

        //Åbn forbindelsen
        objConn.Open();

        //Udfør SQL-kommandoen (en UPDATE/INSERT/DELETE) der ikke giver et
resultatsæt men derimod et tal, nemlig antallet af påvirkede rækker
        CMD.ExecuteNonQuery();
        objConn.Close();

        }
    // ------------------------------------------------------------------

    public DataTable GetDataFactories(MySqlCommand CMD, MySqlConnection CON)
    {

        //Ny MySqlDataAdapter
```

```
        MySqlDataAdapter objDA = new MySqlDataAdapter();

        //Nyt Dataset
        DataSet objDS = new DataSet();

        //Angivelse af MySqlCommando'ens forbindelse
        CMD.Connection = CON;

        //Angivelse af DataAdapterens Select-kommando (da getData kun bruges ved
SELECT-statemants)
        objDA.SelectCommand = CMD;
        objDA.Fill(objDS);

        return objDS.Tables[0];
    }

    public void ModifyDataFactories(MySqlCommand CMD, MySqlConnection CON)
    {
        CMD.Connection = CON;
        CON.Open();
        CMD.ExecuteNonQuery();
        CON.Close();
    }

    // ------------------------------------------------------------------
}
```

Dette var tre små eksempler, der viser, hvordan man laver forbindelse i sit site mellem database og site. Vi vil nu gå i gang med at lave nogle gennerelle funktioner i et website. Til dette vil vi oprette nogle sider, der senere kan bruges som inspiration til mere komplicerede udtræk og dataændringer

Forbindelse til database fra ASP.NET ved hjælp af VB.NET (MySQL)

Vi er nu nået så langt, at vi skal til at forbinde vores .NET-site med vores database.

Man kan lave forbindelsen på 3 måder, som alle vil blive vist i nedenstående eksempler.

Man kan have sin forbindelse stående i web.config-filen. Man kan have den stående i sin dataAccess-fil eller man kan have den stående i sine factories.

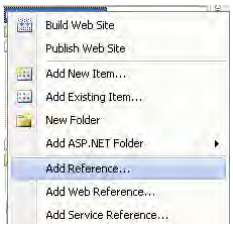
Eksempel på forbindelse via web.config-filen til VB.NET

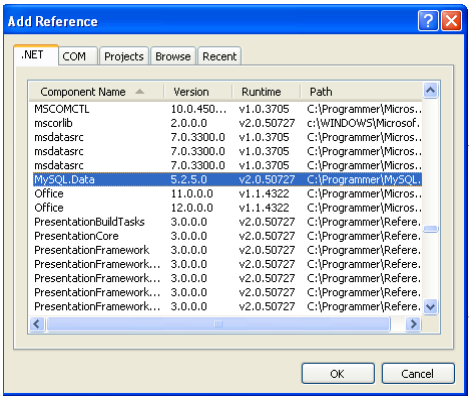
Som udgangspunkt for vores tre små eksempler vil vi bruge den database, som vi har, nemlig dbFrugt.

Vi laver først et eksempel, hvor forbindelsen hentes via web.config-filen.

Man kan i sine sites sagtens have forbindelser til flere forskellige databaser.

Opret et nyt website.





Kald det *startMedWebconfigVB*.
Husk at tilføje en reference til MySQL. Ellers får man ikke adgang til de namespaces, man skal bruge, når ens database er oprettet med MySQL. En reference tilføjes ved at højreklikke på roden i Solution Explorer, vælge punktet Add Reference. Her vælges .NET-fanebladet og MySQL.Data findes. Den vælges og der klikkes OK.

I webconfig-filen under <connectionStrings> skrives en af følgende connectionstrenge alt efter om det er mySQL eller MSSQL man bruger.

```
        <connectionStrings>
                <add name="NavnPaaForbindelse"
connectionString="server=servernavn;database=databasenavn;uid=brugernavn;pwd=pas
sword;"/>
                <add name="NavnPaaForbindelse1"
connectionString="server=Servernavn;database=databasenavn;uid=brugernavn;pwd=pas
word" />
        </connectionStrings>
```

I webconfig-filen kan man som sagt godt have flere forbindelser til forskellige databaser. Her er den første til MySQL og den anden til MSSQL

Opret dernæst dbDataAccess.vb i App_Code-mappen

dbDataAccess.vb

```
Imports Microsoft.VisualBasic
Imports System.Data
Imports MySql.Data.MySqlClient
Imports System.Configuration
```

```
Public Class dbDataAccess

Dim strDB As String =
ConfigurationManager.ConnectionStrings("NavnPaaForbindelse").ConnectionString

'Funktionen til at hente data fra mysql-databasen
    Public Function getData(ByVal CMD As MySqlCommand) As DataTable
        'Ny forbindelse fra Connectionstring'en
        Dim objConn As New MySqlConnection(strDB)
        'Ny MySqlDataAdapter
        Dim da As New MySqlDataAdapter()
        'Nyt Dataset
        Dim ds As New DataSet()

        'Angivelse af MySqlCommando'ens forbindelse
        CMD.Connection = objConn
        'Angivelse af DataAdapterens Select-kommando (da getData kun bruges ved
SELECT-statemants)
        da.SelectCommand = CMD
        da.Fill(ds)

        Return ds.Tables(0)

    End Function

    'Metoden der bruges ved INSERT, UPDATE og DELETE
    'modifyData tager ligeledes en parameter af typen MySqlCommand og returnere
antallet af påvirkede rækker
    Public Sub modifyData(ByVal CMD As MySqlCommand)
        'Forbindelsen
        Dim objconn As New MySqlConnection(strDB)

        'Kommandoens forbindelse
        CMD.Connection = objconn

        'Åbn forbindelsen
        objconn.Open()

        'Udfør SQL-kommandoen (en UPDATE/INSERT/DELETE) der ikke giver et
resultatsæt men derimod et tal, nemlig antallet af påvirkede rækker
        CMD.ExecuteNonQuery()
        objconn.Close()
    End Sub
End Class
```

Klassen her indeholder både en funktion, der kan give adgang til at trække data ud og en subrutine, der giver adgang til at ændre data i databasen.

Opret filen udtraekFac.vb i App_Code-mappen

Lav her en funktion, der trækker alle varenavne ud

```
Imports Microsoft.VisualBasic
Imports System.Data
Imports MySql.Data.MySqlClient
```

```
Public Class udtraekFac

    'giver adgang til de funktioner, der ligger i dbDataAccess.vb
    Dim objConn As New dbDataAccess

    'Funktion, der trækker alt ud fra tabellen tblVarer i databasen dbFrugt
    Public Function getAlleVarer() As DataTable

        'Laver den sql-streng, der skal bruges
        Dim strSQL As String = "Select fldID, fldVarenavn, fldPris, fldAntal
From tblvarer order by fldID desc"

        'bruger CommandObjektet til at gemme resultatsættet i
        Dim objCMD As MySqlCommand = New MySqlCommand(strSQL)

        Return objConn.getData(objCMD)

    End Function
End Class
```

På default.aspx trækkes to literals ind.

Den første skal være til beskrivende tekst, den anden til at vise vores data i.

```
    <form id="form1" runat="server">
    <div>
        <asp:Literal ID="litOverskrift" runat="server"></asp:Literal><br /><br />
        <asp:Literal ID="litVis" runat="server"></asp:Literal>
    </div>
    </form>
```

Default.aspx.vb

```
Imports System.Data

Partial Class _Default
    Inherits System.Web.UI.Page

    Protected Sub Page_Load(ByVal sender As Object, ByVal e As System.EventArgs)
Handles Me.Load
        If Not IsPostBack Then
            litOverskrift.Text = "Her vises alle varenavne i databasen
dbFrugt<br />"
            litOverskrift.Text &= "Vi har her brugt web.config-filen til at huse
forbindelsen til databasen i"

            Dim objUdtraek As New udtraekFac

            For Each row As DataRow In objUdtraek.getAlleVarer().Rows
                litVis.Text &= "Varenavn: " & row.Item("fldVarenavn") & "<br />
vPris: " & row.Item("fldPris") & " kr.<hr /><br />"
            Next
        End If
    End Sub
End Class
```

Eksempel på forbindelse via dbDataAccess.vb til VB.NET

Opret et nyt website. Kald det startMeddbDataAccessVB

I mappen App_Code oprettes filen dbDataAccess.vb

```
Imports Microsoft.VisualBasic
Imports System.Data
Imports MySql.Data.MySqlClient


Public Class dbDataAccess

    ' Klasse til at forbinde til en MySql-database
    ' Husk at importere (Add Reference) MySql.Data

    'Forbindelsen til databasen
    Dim strCon As String =
"Server=servernavn;Database=databasenavn;Uid=brugeridentifikation;Pwd=password;"

    'Funktionen til at hente data fra mysql-databasen
    Public Function getData(ByVal CMD As MySqlCommand) As DataTable
        'Ny forbindelse fra Connectionstring'en
        Dim objConn As New MySqlConnection(strCon)
        'Ny MySqlDataAdapter
        Dim da As New MySqlDataAdapter()
        'Nyt Dataset
        Dim ds As New DataSet()

        'Angivelse af MySqlCommando'ens forbindelse
        CMD.Connection = objConn
        'Angivelse af DataAdapterens Select-kommando (da getData kun bruges ved
SELECT-statemants)
        da.SelectCommand = CMD
        da.Fill(ds)

        Return ds.Tables(0)

    End Function

    'Metoden der bruges ved INSERT, UPDATE og DELETE
    'modifyData tager ligeledes en parameter af typen MySqlCommand og returnere
antallet af påvirkede rækker
    Public Sub modifyData(ByVal CMD As MySqlCommand)
        'Forbindelsen
        Dim objconn As New MySqlConnection(strCon)

        'Kommandoens forbindelse
        CMD.Connection = objconn

        'Åbn forbindelsen
        objconn.Open()

        'Udfør SQL-kommandoen (en UPDATE/INSERT/DELETE) der ikke giver et
resultatsæt men derimod et tal, nemlig antallet af påvirkede rækker
        CMD.ExecuteNonQuery()
        objconn.Close()
    End Sub
End Class
```

Opret også filen udtraekFac.vb i mappen App_Code

```
Imports Microsoft.VisualBasic
Imports System.Data
Imports MySql.Data.MySqlClient

Public Class udtraekFac

    'giver adgang til de funktioner, der ligger i dbDataAccess.vb
    Dim objConn As New dbDataAccess

    'Funktion, der trækker alt ud fra tabellen tblVarer i databasen dbFrugt
    Public Function getAlleVarer() As DataTable

        'Laver den sql-streng, der skal bruges
        Dim strSQL As String = "Select fldID, fldVarenavn, fldPris, fldAntal
From tblvarer order by fldID desc"

        'bruger CommandObjektet til at gemme resultatsættet i
        Dim objCMD As MySqlCommand = New MySqlCommand(strSQL)

        Return objConn.getData(objCMD)

    End Function
End Class
```

På default.aspx trækkes to literals ind.

Den første skal være til beskrivende tekst, den anden til at vise vores data i.

```
    <form id="form1" runat="server">
    <div>
        <asp:Literal ID="litOverskrift" runat="server"></asp:Literal><br /><br />
        <asp:Literal ID="litVis" runat="server"></asp:Literal>
    </div>
    </form>
```

Default.aspx.vb

```
Imports System.Data

Partial Class _Default
    Inherits System.Web.UI.Page

    Protected Sub Page_Load(ByVal sender As Object, ByVal e As System.EventArgs)
Handles Me.Load
        If Not IsPostBack Then
```

```
            litOverskrift.Text = "Her vises alle varenavne i databasen
dbFrugt<br />"
            litOverskrift.Text &= "Her ligger forbindelsen til databasen i
dbDataAccess-filen"

            Dim objUdtraek As New udtraekFac

            For Each row As DataRow In objUdtraek.getAlleVarer().Rows
                litVis.Text &= "Varenavn: " & row.Item("fldVarenavn") & "<br />
Pris: " & row.Item("fldPris") & " kr.<hr /><br />"
            Next

        End If
    End Sub
End Class
```

Eksempel på forbindelse via factories til VB.NET

Opret et nyt website. Kald det startMedFactoriesVB

Opret filen dbDataAccess.vb i mappen App_Code

```
Imports Microsoft.VisualBasic
Imports System.Data
Imports MySql.Data.MySqlClient

Public Class dbDataAccess

    'Funktionen til at hente data fra mysql-databasen
    Public Function GetData(ByVal CMD As MySqlCommand, ByVal CON As
MySqlConnection) As DataTable

        'Nyt Dataset
        Dim objDS As New DataSet
        'Ny MySqlDataAdapter
        Dim objDA As New MySqlDataAdapter()

        'Angivelse af MySqlCommando'ens forbindelse
        CMD.Connection = CON

        'Angivelse af DataAdapterens Select-kommando (da getData kun bruges ved
SELECT-statemants)
        objDA.SelectCommand = CMD
        objDA.Fill(objDS)

        Return objDS.Tables(0)
    End Function

    'Metoden der bruges ved INSERT, UPDATE og DELETE
    'modifyData tager ligeledes en parameter af typen MySqlCommand og returnerer
antallet af påvirkede rækker
    Public Sub ModifyData(ByVal CMD As MySqlCommand, ByVal CON As
MySqlConnection)
        'Kommandoens forbindelse
        CMD.Connection = CON

        'Åbn forbindelsen
```

```
        CON.Open()

        'Udfør SQL-kommandoen (en UPDATE/INSERT/DELETE) der ikke giver et
resultatsæt men derimod et tal, nemlig antallet af påvirkede rækker
        CMD.ExecuteNonQuery()

        'Luk forbindelsen
        CON.Close()

    End Sub
End Class
```

Klassen her indeholder både en funktion, der kan give adgang til at trække data ud og en subrutine, der giver adgang til at ændre data i databasen.

Opret også filen udtraekFac.vb i mappen App_Code

```
Imports Microsoft.VisualBasic
Imports System.Data
Imports MySql.Data.MySqlClient

Public Class udtraekFac

    'forbindelse til database
    Dim strCon As String = "Server=servernavn;Database=
databasenavn;Uid=brugeridentifikation;Pwd=password;”

    Dim _Con As New MySqlConnection(strCon)

    Dim objConn As New dbDataAccess
    'Funktion, der trækker alt ud fra tabellen tblVarer i databasen dbFrugt
    Public Function getAlleVarer() As DataTable

        'Laver den sql-streng, der skal bruges
        Dim strSQL As String = "Select fldID, fldVarenavn, fldPris, fldAntal
From tblvarer order by fldID desc"

        'bruger CommandObjektet til at gemme resultatsættet i
        Dim objCMD As MySqlCommand = New MySqlCommand(strSQL)

        Return objConn.getData(objCMD, _Con)


    End Function
End Class
```

På default.aspx trækkes to literals ind.

Den første skal være til beskrivende tekst, den anden til at vise vores data i.

```
    <form id="form1" runat="server">
    <div>
        <asp:Literal ID="litOverskrift" runat="server"></asp:Literal><br /><br />
        <asp:Literal ID="litVis" runat="server"></asp:Literal>
    </div>
    </form>
```

Default.aspx.vb

```
Imports System.Data

Partial Class _Default
    Inherits System.Web.UI.Page

    Protected Sub Page_Load(ByVal sender As Object, ByVal e As System.EventArgs)
Handles Me.Load
        If Not IsPostBack Then

            litOverskrift.Text = "Her vises alle varenavne i databasen
dbFrugt<br />"
            litOverskrift.Text &= " Her ligger forbindelsen til databasen i de
enkelte factories"

            Dim objUdtraek As New udtraekFac

            For Each row As DataRow In objUdtraek.getAlleVarer().Rows
                litVis.Text &= "Varenavn: " & row.Item("fldVarenavn") & "<br />
Pris: " & row.Item("fldPris") & " kr.<hr /><br />"
            Next

        End If
    End Sub
End Class
```

Man kan selvfølgelig sagtens samle de forskellige forbindelser i en fil (til VB.NET)

dbDataAccessSamlet.vb

```
Imports Microsoft.VisualBasic
Imports System.Data
Imports MySql.Data.MySqlClient
Imports System.Configuration

Public Class dbDataAccessSamlet

    'forbindelse via WebConfig

    Dim strDB As String =
ConfigurationManager.ConnectionStrings("NavnPaaForbindelse").ConnectionString

    '--------------------------------------


    'Funktionen til at hente data fra mysql-databasen
    Public Function getDataWebConfig(ByVal CMD As MySqlCommand) As DataTable
        'Ny forbindelse fra Connectionstring'en
        Dim objConn As New MySqlConnection(strDB)
        'Ny MySqlDataAdapter
        Dim da As New MySqlDataAdapter()
        'Nyt Dataset
        Dim ds As New DataSet()
```

```
        'Angivelse af MySqlCommando'ens forbindelse
        CMD.Connection = objConn
        'Angivelse af DataAdapterens Select-kommando (da getData kun bruges ved
SELECT-statemants)
        da.SelectCommand = CMD
        da.Fill(ds)

        Return ds.Tables(0)
    End Function

    'Metoden der bruges ved INSERT, UPDATE og DELETE
    'modifyData tager ligeledes en parameter af typen MySqlCommand og returnerer
antallet af påvirkede rækker
    Public Sub modifyDataWebConfig(ByVal CMD As MySqlCommand)
        'Forbindelsen
        Dim objconn As New MySqlConnection(strDB)

        'Kommandoens forbindelse
        CMD.Connection = objconn

        'Åbn forbindelsen
        objconn.Open()

        'Udfør SQL-kommandoen (en UPDATE/INSERT/DELETE) der ikke giver et
resultatsæt men derimod et tal, nemlig antallet af påvirkede rækker
        CMD.ExecuteNonQuery()
        objconn.Close()
    End Sub

    '--------------------------------------

    'forbindelse via dbDataAccess-filen

    Dim strCon As String = "Server=servernavn;Database=
databasenavn;Uid=brugeridentifikation;Pwd=password;"

    'Funktionen til at hente data fra mysql-databasen
    Public Function getDatadbConn(ByVal CMD As MySqlCommand) As DataTable
        'Ny forbindelse fra Connectionstring'en
        Dim objConn As New MySqlConnection(strCon)
        'Ny MySqlDataAdapter
        Dim da As New MySqlDataAdapter()
        'Nyt Dataset
        Dim ds As New DataSet()

        'Angivelse af MySqlCommando'ens forbindelse
        CMD.Connection = objConn
        'Angivelse af DataAdapterens Select-kommando (da getData kun bruges ved
SELECT-statemants)
        da.SelectCommand = CMD
        da.Fill(ds)

        Return ds.Tables(0)
    End Function

    'Metoden der bruges ved INSERT, UPDATE og DELETE
    'modifyData tager ligeledes en parameter af typen MySqlCommand og returnerer
antallet af påvirkede rækker
    Public Sub modifyDatadbConn(ByVal CMD As MySqlCommand)
```

```
        'Forbindelsen
        Dim objconn As New MySqlConnection(strCon)

        'Kommandoens forbindelse
        CMD.Connection = objconn

        'Åbn forbindelsen
        objconn.Open()

        'Udfør SQL-kommandoen (en UPDATE/INSERT/DELETE) der ikke giver et
resultatsæt men derimod et tal, nemlig antallet af påvirkede rækker
        CMD.ExecuteNonQuery()
        objconn.Close()
    End Sub

    '--------------------------------------
    'forbindelse via factories
    'Funktionen til at hente data fra mysql-databasen
    Public Function getDataFactories(ByVal CMD As MySqlCommand, ByVal CON As
MySqlConnection) As DataTable

        'Nyt Dataset
        Dim objDS As New DataSet
        'Ny MySqlDataAdapter
        Dim objDA As New MySqlDataAdapter()

        'Angivelse af MySqlCommando'ens forbindelse
        CMD.Connection = CON

        'Angivelse af DataAdapterens Select-kommando (da getData kun bruges ved
SELECT-statemants)
        objDA.SelectCommand = CMD
        objDA.Fill(objDS)

        Return objDS.Tables(0)
    End Function

    'Metoden der bruges ved INSERT, UPDATE og DELETE
    'modifyData tager ligeledes en parameter af typen MySqlCommand og returnerer
antallet af påvirkede rækker
    Public Sub modifyDataFactories(ByVal CMD As MySqlCommand, ByVal CON As
MySqlConnection)
        'Kommandoens forbindelse
        CMD.Connection = CON

        'Åbn forbindelsen
        CON.Open()

        'Udfør SQL-kommandoen (en UPDATE/INSERT/DELETE) der ikke giver et
resultatsæt men derimod et tal, nemlig antallet af påvirkede rækker
        CMD.ExecuteNonQuery()

        'Luk forbindelsen
        CON.Close()
    End Sub
End Class
```

Dette var tre små eksempler, der viser, hvordan man laver forbindelse i sit site mellem database og site. Vi vil nu gå i gang med at lave nogle gennerelle funktioner i et website. Til dette vil vi oprette nogle sider, der senere kan bruges som inspiration til mere komplicerede udtræk og dataændringer

Lidt avanceret SQL

Man kan lave rigtig mange spændende ting med SQL, f. eks. deludtræk og beregninger.
Her kommer et par eksempler. Der skal gøres opmærksom på at det ikke er udtræk fra tabeller, der tidligere er brugt i materialet. Men de kan tilrettes og man kan få inspiration af dem.

Hvis man nu kun vil have valgt række 2 og 3 fra en tabel med mange rækker. Kan man det? Ja det kan man godt.

Sql-en får dette udseende:

```
select top 2 fldOverskrift, fldTekst from tblBlog where fldID <>(select top 1
fldID from tblBlog order by fldID desc) order by fldID desc
```

Har man et date-felt og her kun vil have månedsnavnet trukket ud, så kan det også lade sig gøre

```
select  distinct DatePart(mm,fldOprettet) as dato from tblBlog
```

Man bruger her DatePart. fldOprettet er feltnavnet og det er af typen Date. Det reserverede ord distinct sørger for at hver måned kun bliver vist en gang.

Resultatet vises som tal og man kan så i sin ASP.net kode lave en Switch Case, der kan vise månedsnavne i stedet for.

Beregninger
I denne sql-sætning skal man bemærke round, der viser, hvor mange decimaler et resultat skal vises med og cast, som tvinger et tal til at være af en anden datatype (her fra int til float)

```
Select Sum(tblArtsOpl.fldAntal) as antalFugle,
round(sum(tblArtsOpl.fldAntal)/cast(97 as float)*100, 2) as Procent,
tblMaaneder.fldMaaned as Maaned from tblArtsOpl inner join tblMaaneder on
tblArtsOpl.fldMaanedID = tblMaaneder.fldID where tblArtsOpl.fldArtsID=3 group by
tblArtsOpl.fldArtsID, tblMaaneder.fldMaaned, tblArtsOpl.fldMaanedID
```

QUIZZ

Hvorfor skal en tabel være normaliseret?

Hvilken rækkefølge skal følgende komme i?

ORDER BY
GROUP BY
SELECT
HAVING
WHERE
FROM

Skal der altid være en select i et udtræk fra en tabel?

Ja
Nej

Er der fejl i følgende SQL-Statement?
*SELECT fldVarenavn, fld Pris, (fldPris * fldAntal) as 'SUM' ORDER BY fldVarenavn*

Nævn de måder man kan lave en forbindelse mellem sit site og databasen

Hvad bruges update til?
Hvad bruges delete til?
Hvad bruges insert til?

Hvor skal LIMIT stå i et udtræk?

Hvad gør man med LIMIT?

Hvad gør man med TOP?

Er LIMIT et ord, der kan bruges i alle databaser?

Vis hvordan TOP bruges

Er følgende korrekt
Select LIMIT 3 fldVarenavn, fldPris from tblVarer