Første undervisningsgang

Database

Noter til dette slideshow

- Slideshowet er gjort tilgængeligt for jer så I kan følge det på jeres egen pc og ikke kun på projektoren
- De vigtigste nøgleord er markeret med støvet blå
- Andet, især fagtermer, der er værd at læse op på, er markeret med lilla
- Bagerst i slideshowet er der en kilde sektion, hvor man finder links til noget af det materiale der danner grundlag for denne lektion
- Almindelige slides har støvet blå top-bar, slides med opgaver har lilla top-bar

Denne undervisningsgang

- Introduktion
  - Kurset
  - Underviseren
  - Jer selv
  - Stedet
  - Fronter
  - Lektionsplan
- Konceptet bag databaser
  - Hvorfor
  - Forskellige typer databaser
    - Flad fil databaser
    - Relationelle databaser
    - Hierarkiske databaser
- ER diagrammer
- Grundlæggende datastrukturering
- Introduktion til MySQL Workbench

Introduktion

Lidt om noget af det forskellige bag kurset

Introduktion - Kurset

- Vi benytter bogen ”**S**” fra S som grundlag
  - Planen er at vi arbejder os igennem en del af bogen i løbet af kurset
- Udover bogen benytter vi videoer med tilhørende tests fra Microsoft Virtual Academy
- Målet er at have grundlæggende færdigheder inden for
  - Planlægning af produktion af en program
  - Grundlæggende forståelse af teorien bag konceptet programmering
  - Grundlæggende færdigheder inden for programmering i sproget C#

Introduktion - Underviseren

- **Erik Weber-Lauridsen**
- erwl@eal.dk
- Vokset op med IT
- Programmeret siden han var 11
- Oprindeligt folkeskolelærer
- Bachelorgrad i webudvikling
- Underviser på erhvervsskole
  - Web-integrator og medie-grafiker
- Underviser på erhvervsakademi
  - Multimedia designer og bachelor i webudvikling
- Hvis I vil se noget af det andet jeg har undervist i så tjek iul.dk

Introduktion - Navnerunde

- Præsenter jer selv:
  - Navn
  - Job / uddannelse
    - Uddyb gerne med hvad I finder særligt interessant ved jeres fag for tiden
  - Hvilken for jer unik personlig data ville I gemme i en database?

  - Eksempel
    - Erik
    - Adjunkt
      - VR, AR og mobiludvikling
    - Tegneserier

Introduktion - Stedet

- Ledelsesakademiet
  - Erhvervs Akademiet Lillebælt
- Reception
- Undervisningslokaler
- Toiletter
- Kantine
- Administration
- Rundvisning

Introduktion - Fronter

- Fronter er det system vi her på stedet bruger til at dele filer og information gennem
- https://fronter.com/ledelsesakademiet/
- På forsiden ses opdateringer fra alle de rum (hold) man er på
- Under rum kan man se de hold man er på
- Hvis man klikker på et hold åbnes det i en ny fane
  - Klik på venstre side af fanebladet for at pinne det, så fanen altid er åben når du går på Fronter
- Under rummet kan man se seneste nyt osv. på dets forside
- Du finder lektionsplaner og filer til de enkelte lektioner under Materiale
- Jeg regner ikke med at anvende Portfolio funktionen

Introduktion - Lektionsplan

- Lektionsplanen findes som sagt på Fronter under arkiv
- Den er *ikke* sat i sten
  - Den kan blive ændret undervejs baseret på ønsker og behov der måtte vise sig
  - Derfor er den versionsnummereret så I kan sikre jer at I altid har den nyeste udgave

Konceptet bag databaser

Hvorfor bruge databaser

Hvorfor

- Databaser er en meget effektiv metode til håndtering af store mængder forskellige typer data med lethed.
- Databaser gør det muligt at gemme data systematisk, og disse data kan nemt hentes, filtreres, sorteres og opdateres effektivt og præcist.

Forskellige typer databaser
Flad Fil databaser

Flad fil databaser er et enkelt databasesystem, der lagrer optegnelser i en almindelig tekstfil, der ikke har strukturerede forhold mellem hver post.

Brug

- Ideel til opbevaring af meget små mængder simple data, der kan håndteres manuelt

Forskellige typer databaser
Flad Fil databaser

Kendetegn og features

- Gemmer al data i en stor tabel
- Hver linje i teksten indeholder en post
- Den første række i en flad fil refererer til feltnavnet
- De forskellige felter i en post adskilles af afgrænsere, såsom lodret bjælke "|" eller et komma "," eller et semi kolon ";”
- Ingen mapper eller stier bruges til at organisere data
- Kan ikke gemme grafiske dokumenter, men kun tekst
- Data, der er gemt deri, kan søges ved hjælp af søgeord, sætninger eller begge dele

Forskellige typer databaser
Flad Fil databaser

Eksempel på brug

- Vi kan bruge flad fildatabase til at gemme registreringer af finansiering eller besparelser til en lille gruppe mennesker, fordi der ikke er mange felter, brug af flere tabeller er heller ikke nødvendig, og der er ingen kompleks processer at udføre

SEMESTER 3 Fundings									
Date	Amount		AMT	BNA	MWTS	PHH	SSMN	SY	ZPH	Note
18/09/2015	K	6,000		√	√	√	√	√	√	
25/09/2015	K	6,000		√	√	√	√	√	√	
2/10/2015	K	7,000	√	√	√	√	√	√	√	
9/10/2015	K	7,000	√	√	√	√	√	√	√	
16/10/2015	K	7,000	√	√	√	√	√	√	√	
16/10/2015	K	3,000	√	√					√	Fine money
23/10/2015	K	7,000	√	√	√	√	√	√	√	
30/10/2015	K	7,000	√	√	√	√	√	√	√	
6/11/2015	K	6,000	√	√	√		√	√	√	
13/11/2015	K	7,000	√	√	√	√	√	√	√	
20/11/2015	K	5,000	√		√		√	√	√	
27/11/2015	K	3,000	√					√	√	
18/12/2015	K	2,000	√						√	
25/12/2015	K	1,000	√							
1/1/2016	K	1,000	√							
8/1/2016	K	1,000	√							
Total	K	76,000								

Forskellige typer databaser
Flad Fil databaser

Eksempel på formater

- csv filer

```
Year,Make,Model,Description,Price
1997,Ford,E350,"ac, abs, moon",3000.00
1999,Chevy,"Venture ""Extended Edition""","",4900.00
1999,Chevy,"Venture ""Extended Edition, Very Large""",,5000.00
1996,Jeep,Grand Cherokee,"MUST SELL!
air, moon roof, loaded",4799.00
```

Year	Make	Model	Description	Price
1997	Ford	E350	ac, abs, moon	3000.00
1999	Chevy	Venture "Extended Edition"		4900.00
1999	Chevy	Venture "Extended Edition, Very Large"		5000.00
1996	Jeep	Grand Cherokee	MUST SELL! air, moon roof, loaded	4799.00

Forskellige typer databaser
Flad Fil databaser

Eksempel på formater

- ini filer
  - INI-filformatet er en uformel standard for konfigurationsfiler til nogle platforme eller software. INI-filer er enkle tekstfiler med en grundlæggende struktur sammensat af sektioner, egenskaber og værdier.

```
; last modified 1 April 2001 by John Doe
[owner]
name=John Doe
organization=Acme Widgets Inc.

[database]
; use IP address in case network name resolution is not working
server=192.0.2.62
port=143
file="payroll.dat"
```

Forskellige typer databaser
Flad Fil databaser

Fordele

- Nemmere at installere og bruge
- Kræver mindre plads
- Ingen specielle software- eller hardwarekrav
- Ofte gratis eller billig

Note

- XML og JSON er *ikke* flad-fil databaser da deres struktur danner et hierarki

Ulemper

- *Ingen* form for indeksering
- Sårbar over for data korruption eller duplikering
- Tilbøjelig til fejl
- Svær at opdatere eller ændre
- Dårlig adgangskontrol
- Kan ikke udføre komplekse processer

Forskellige typer databaser
Relationelle databaser

Mere avanceret og effektiv type database, der kan gemme meget store mængder data i sæt af tabeller, der er sammenkædet

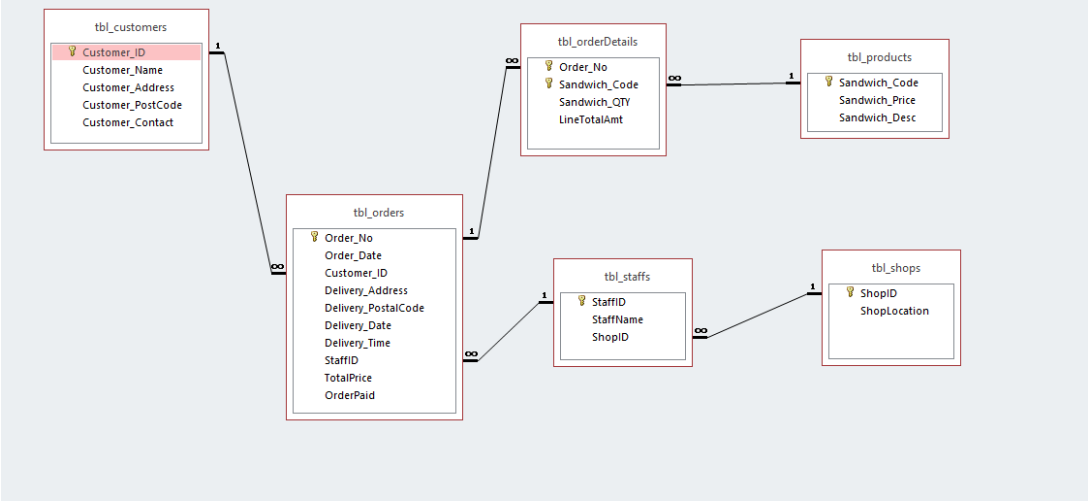
Kendetegn og features

- Brug af flere tabeller til at gemme poster og hver tabel består af poster i rækker og kolonner
- Hver kolonne er et felt, der repræsenterer en bestemt type information om enheden, og hver række fungerer som en post
- Hvert felt i en tabel har sin egen datatype
- Hver række indeholder en unik forekomst af data, der unikt identificerer en post
- Optegnelser i tabellerne er knyttet til poster i andre tabeller gennem et forhold
- Opgørelserne om at indsætte, hente, ajourføre og slette data i relationelle databaser er lavet af forespørgsler, som er skrevet i SQL

Forskellige typer databaser
Relationelle databaser

Eksempel på brug

- Udbredt i mange forskellige brancher fra små til store virksomheder til at:
  - lagre regnskaber for hele branchen
  - holde styr på inventar
  - hold kunde og leverandør oplysninger
  - holde styr på kundeordrer
  - holde optegnelser over medarbejdere osv.



Forskellige typer databaser
Relationelle databaser

Fordele

- Kan gemme stor mængde data
- Sikrer dataintegritet
- Let udvidelig og modificerbar
- Tilrettelægger datatilgænge-lighed, søgbarhed og rappor-tering
- Bedre præstation
- Tillad flere brugere
- Avanceret datasikkerhed

Ulemper

- Stejl indlæringskurve
- Dyr at oprette og vedligeholde
- Kræv sofistikerede hardware og netværksopsætninger

Kommentar

- Relations modellen for data blev foreslået af E.F. Codd i 1970

Forskellige typer databaser
Hierakiske databaser

Database type hvor data er organiseret i en træstruktur, der forbinder et antal forskellige elementer til en "forælder" primærpost

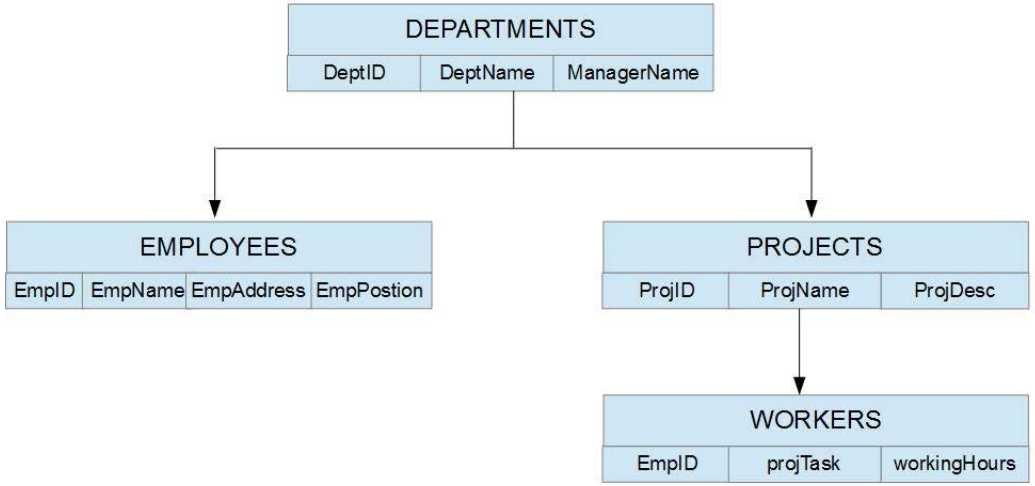
Kendetegn og features

- Øverst på træet er forældrene, og grenene er børn
- Alle attributter af en bestemt post er angivet under en enhedstype (forælder)
- En entitetstype svarer til en tabel
- Hver enkelt post er repræsenteret som en række og en attribut som en kolonne
- Hver optagelsestype har kun en forælder
- Entitetstyper er relateret til hinanden ved hjælp af et til mange forhold

Forskellige typer databaser
Hierakiske databaser

Eksempel på brug

- Mest relevant til brug i den situation, hvor det primære fokus på information er samlet på et enkelt dataelement, såsom en liste over forretnings-afdelinger, medarbejder-organisation i virksomheder eller aktiver
- For eksempel kan vi bruge det hierarkiske skema for en del af FIRMA-databasen



Forskellige typer databaser
Hierakiske databaser

Fordele

- Nemmere at forstå
- Forenkler dataoversigten
- Nemmere at arbejde med på grund af sin lineære form for datalagring

Ulemper

- Prædefineret træstruktur reducerer fleksibiliteten
- Mange til mange relationer understøttes ikke
- Langsom, fordi adgang til et barne-segment kun kan ske gennem modersegmentet

Hierarchical Databases

- Disadvantages
  - Lousy performance with large data collection
    - Inefficient navigation in addition to bloated data
  - VERY expensive
    - $ Millions on mainframes
    - Large staff required to maintain system
  - Specialized skills required to obtain information
    - Experts controlled access to data
      - » inefficient to the speed of business

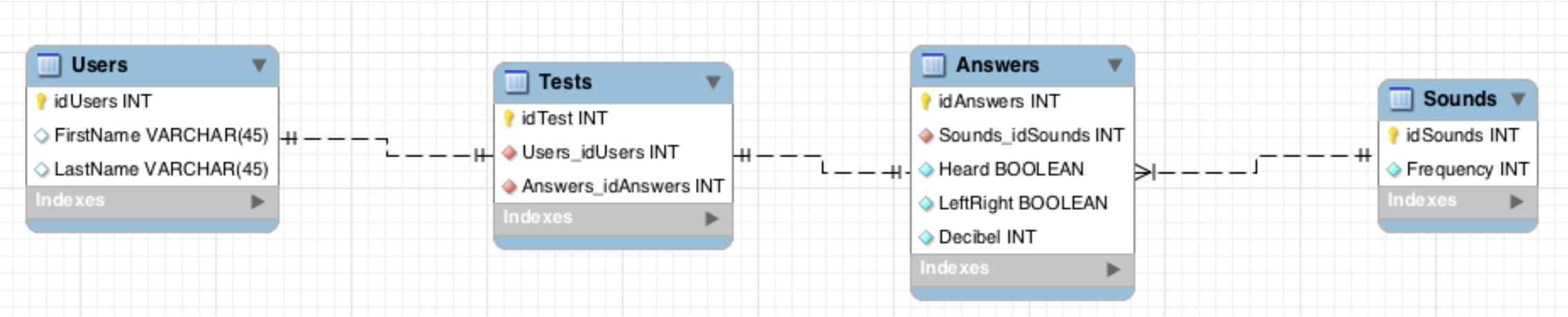
ER diagrammer

Diagrammer der grafisk viser og tydeliggør relationelle datastrukturer

Vi dækker samtidig relationer i databaser

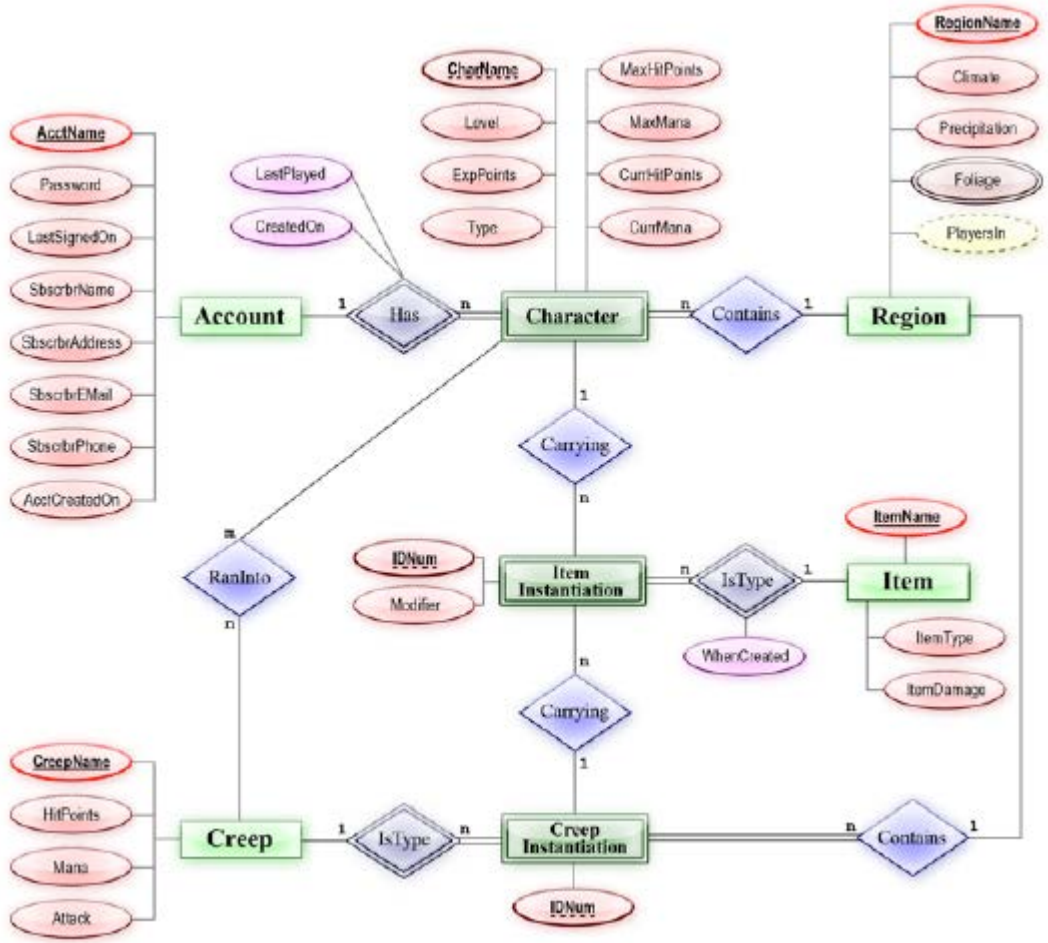
Hvad er et ER diagram

- Et E/R diagram (entity relationship diagram) er en visuel gengivelse af data og hvordan disse relaterer til hinanden.
- Diagrammet illustrerer de forskellige tabeller i databasen, bestående af rækker af kolonner med en dertilhørende primær-nøgle.



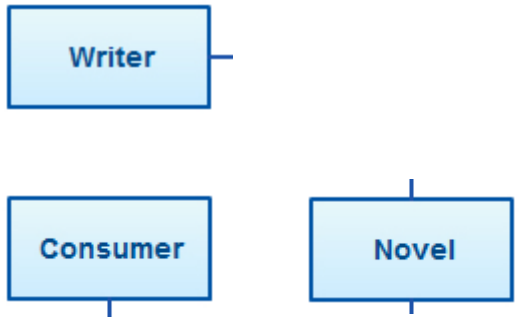
Historien bag ER diagrammer

- Denne ensartede diagram type blev udviklet af Peter Chen i 1976
- Det oprindelige formål var på en systematisk måde at beskrive og definere en forretningsproces
- Chens notation så dog lidt anderledes ud end den vi oftest ser i dag (se f.eks. Sidste slide):



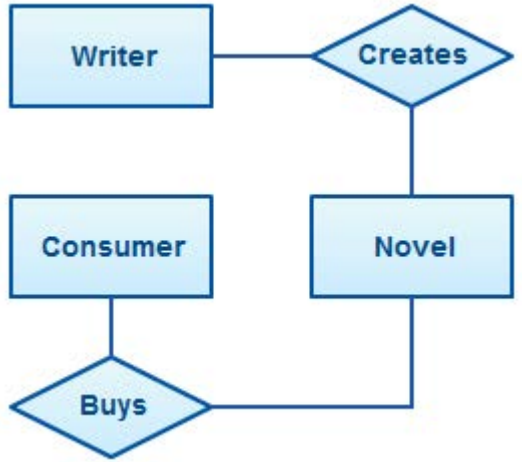
Hvad er en Entity

- En entity kan defineres som et element der har en uafhængig eksistens og kan entydigt identificeres
- En entity er et element der eksisterer enten fysisk eller logisk
- En entity kan betragtes som et navneord
  - en computer, en medarbejder, en sang, et matematisk udtryk



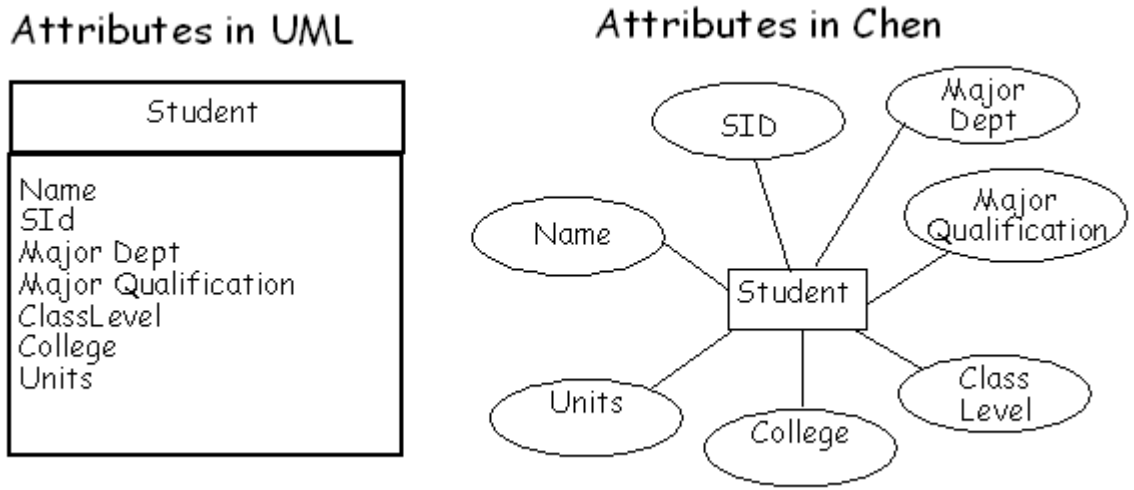
Entiteters forhold

- Et forhold er hvordan enheder er relateret til hinanden
- Forhold kan betragtes som verber (udsagnsord), der knytter to eller flere navneord sammen
  - Et ejerforhold mellem et selskab og en computer
  - Et superviser forhold mellem en medarbejder og en afdeling,
  - En udfører (performs) forhold mellem en kunstner og en sang
  - Et bevist forhold mellem en matematiker og et bevis



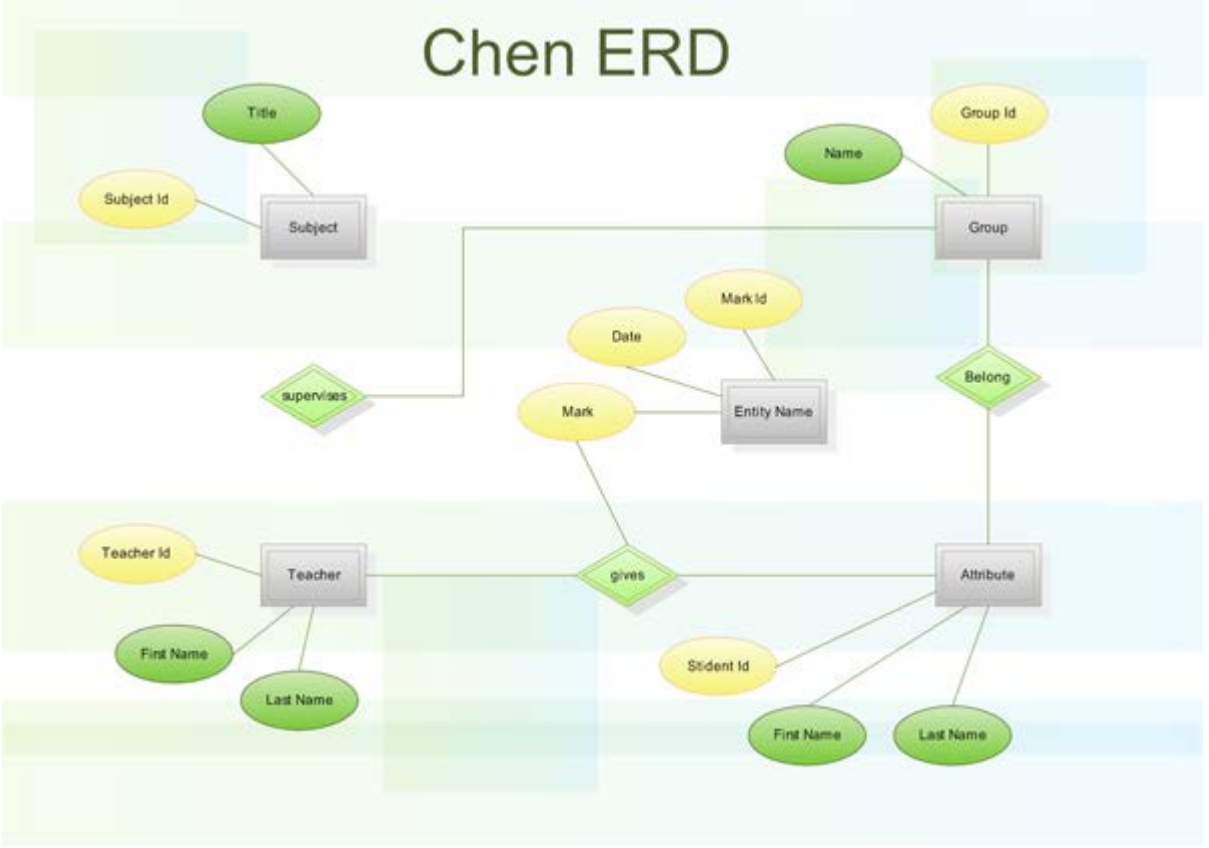
Attributter / egenskaber

- Entities og relationer kan begge have attributter. Eksempler:
  - En medarbejder entity kan have en CPR-nummer attribut. Det afledte forhold kan have en dato attribut
- Hver entity skal have et minimalt sæt af entydigt identificerende attributter /egenskaber, som kaldes entitetens primære nøgle



Chens notation

- Entity/enheds-relation modellering bruger rektangler til at repræsentere enhedssæt, og diamanter til at repræsentere forhold, der passer til første klasse objekt: de kan have attributter og deres egne forhold.
- Hvis et enhedssæt deltager i et forholds sæt er de er forbundet med en linje.

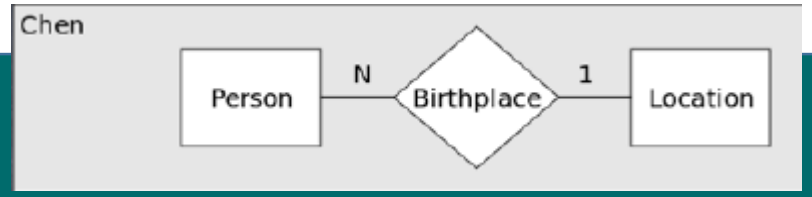


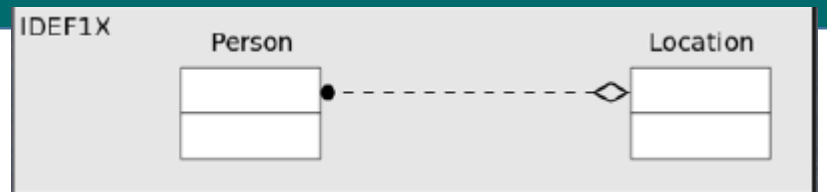
Chens notation

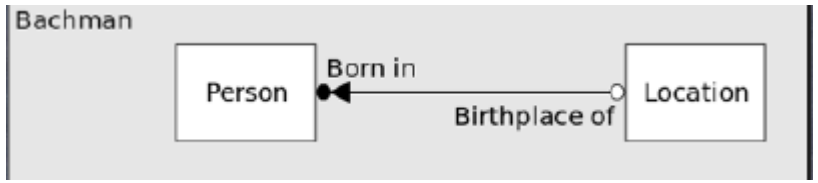
- En dobbelt linje angiver en deltagelsesbegrænsning eller totalitet: alle enheder i enhedens sæt skal deltage i mindst et forhold i forholds sættet
- En pil fra en enhed til et forholdssæt angiver en nøglebegrænsning, dvs. injiceringsevne: hver enhed af entitets sættet kan deltage i max ét forhold i forholds-sættet
- En tyk linje indikerer begge, dvs. bijektivitet: hver enhed i entiteten er sæt involveres i nøjagtigt et forhold.
- Et understreget navn på en attribut indikerer, at det er en nøgle: to forskellige enheder eller relationer med denne attribut vil altid haveforskellige værdier for denne attribut.

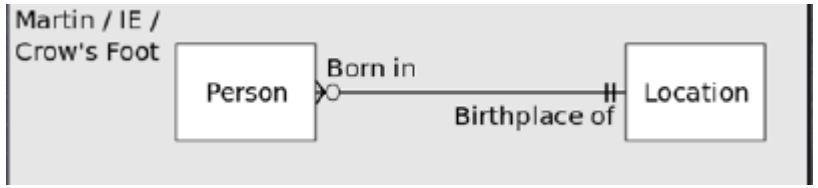
Notation

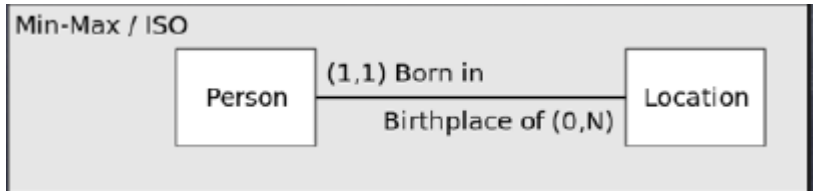
- Der er mange muligheder at vælge imellem.
- Mens crows foot notation er bredt accepteret som den mest intuitive stil, foretrækker nogle udviklere OMT, IDEF, Bachman, eller UML notation for at indikere kardinalitet.
- Crows foot notation viser både minimum og maksimal kardinalitet i et let læseligt grafisk format og er meget udbredt

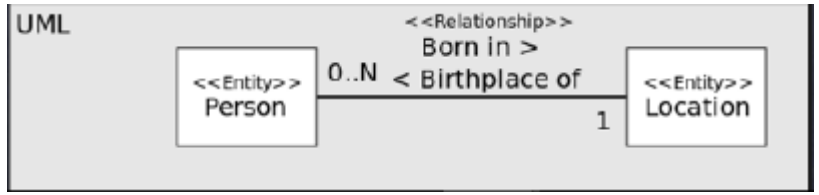












Crow's Foot notation

- Enheder vises som kasser og relationer som linjer mellem kasser
- Forskellige former i enderne af disse linjer repræsenterer kardinalitet af forholdet
- Eksempel
  - Et valgfrit forhold vises mellem Kunstner og sang
  - Symbolerne tættest på sang-enheden repræsenterer at en kunstner kan have "nul, en eller mange" sange, mens sang har "én og kun én "kunstner.
  - Det læses derfor som at en kunstner (kan) udføre nul, en, eller mange sang(e).

Notation: Entities / enheder

- Stærke enheder eksisterer uafhængigt af andre enheder.
  De besidder altid en eller flere attributter, der entydigt skelne hver forekomst af entiteten.

- Svage enheder afhænger af en anden entitetstype. De gør at de ikke besidder unikke attributter (også kendt som en primærnøgle) og ingen mening har i diagrammet uden at være afhængige af en en anden enhed. Denne anden enhed er kendt som ejer.
- Associative enheder er enheder, der forbinder forekomsterne af en eller flere enhedstyper. De indeholder også attributter som er unikke for forholdet mellem de pågældende instanser.

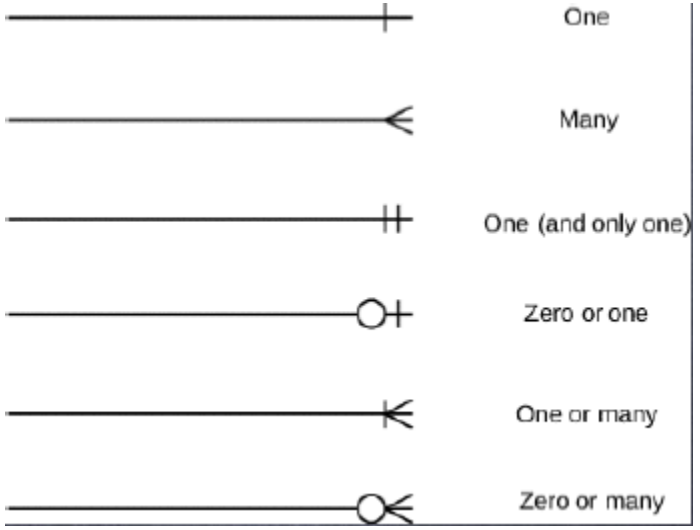
Notation: Forhold

- Forhold er meningsfulde foreninger mellem eller blandt enheder
- De er normalt verber, f.eks. tildele, associere eller spore
- Et forhold giver nyttige oplysninger, der ikke kan skelnes ud fra entitetstyperne alene
- Svage relationer eller identificerende relationer, er forbindelser der eksisterer mellem en svag enheds type og dens ejer.

- Relationer illustrerer en forening mellem to tabeller.
- I den fysiske datamodel er relationer er repræsenteret af stiliserede linjer.

Notation: Forhold

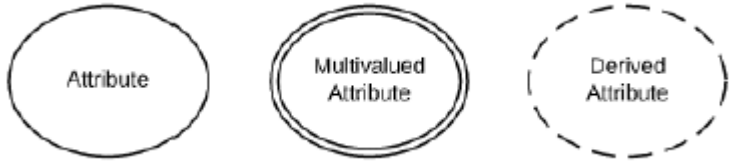
- Kardinalitet og ordinalitet, henviser henholdsvis til maksimum antal gange en forekomst i en enhed kan være forbundet med forekomster i den tilknyttede enhed og det mindste antal gange en forekomst i en enhed kan være forbundet med en forekomst i relateret enhed.



- Kardinalitet og ordinalitet er repræsenteret af styling af en linje og dens endepunkt, som angivet af den valgte notations stil.

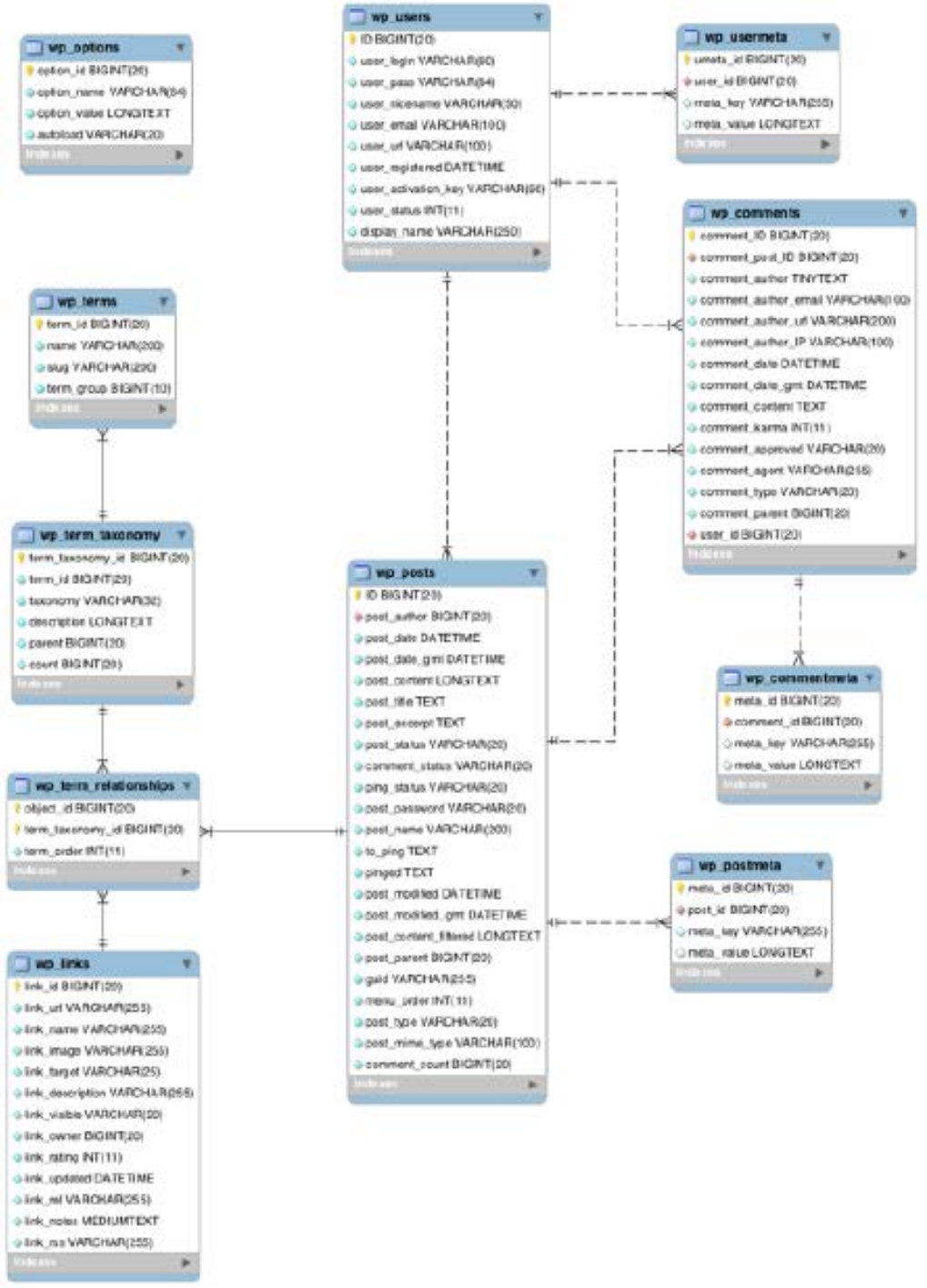
Notation - Attributter

- Attributter er egenskaber for enten en enhed, et mange til mange forhold eller et en-til-en forhold



- Mange-værdi attributter er dem der er i stand til at indeholde mere end en værdi
- Afledte attributter er attributter, hvis værdi kan beregnes ud fra relaterede attributværdier.

Eksempel: WordPress' officielle ER diagram



CRUD

Basal læsning, skrivning og sletning

Hvad er CRUD

- Det står for Create, Read, Update, Delete
- I SQL bruges queries til dette
- I SQL er der mange reseverede ord, flere af dem kan ikke anvendes som tabel navne f.eks. kan man ikke kalde en tabel for "table", "drop" eller "delete".
  - Reseverede ord er typisk angivet med blå tekst i en query og det vil være en fordel, at undgå disse reseverede ord som tabel- og kolonne navne.

- Hvis I vil teste SQL kan I enten bruge MySQL Workbench eller http://sqlfiddle.com/

Create

```
INSERT INTO users
(username, email, password)
VALUES
("johndoe", "john@doe.com", “dragon")
```

- Når man indsætter data i en tabel bruger man `INSERT` kommandoen, så navnet på databasen, navnene på rækkerne
- Værdierne kommer efter `VALUE` nøgleordet

Read

```
SELECT username, email FROM users
```

- Man vælger (og dermed finder reelt) data i en database med SELECT nøgleordet

Update

```
UPDATE users SET
     email = "johndoe@gmail.com",
     name = "John Doe"
WHERE id = 12 users
```

- Når man vil opdatere et felt i en database gøre det med en kombination af UPDATE og oftest WHERE for kun at opdatere hvis et kriterie er opfyldt

Delete

```
DELETE FROM users
WHERE id = 12
```

- Når man vil slette data bruges `DELETE` kommandoen
- Hvis man vil slette en helt tabel bruger man dog DROP, men den kommer vi tilbage til i næste lektion

Grundlæggende datastrukturering

Ting man bør tænke over i det man gemmer sin data

- eller måske allerede idet man producerer den

En tabel der har brug for normalisering

- Normalisering betyder at man sorterer data så det ligger i tabeller på den mest optimale måde
- Man sparer hurtigt resurser og tid ved normalisering
  - Man skal dog tænke lidt mere fra start og ikke smide data ind med en skovl
- Der er en række forskellige normalformer, men der er bred enighed om de førte frem

SalesStaff						
employeeID	salesPerson	salesOffice	officeNumber	customer1	customer2	customer3
1003	Mary Smith	Chicago	312-555-1212	Ford	GM	
1004	John Hunt	New York	212-555-1212	Dell	HP	Apple
1005	Martin Hap	Chicago	312-555-1212	Boeing		

Unormaliseret tabel

1st normal form (1NF)

Definition

- En tabel er i sin første normal form (1NF) når og
- kun når:
  - Der er en primær nøgle der unikt identificerer hver række
  - Hver kolonne indeholder en atomar værdi og der ikke er gentaget indhold i kolonnen.

Formål

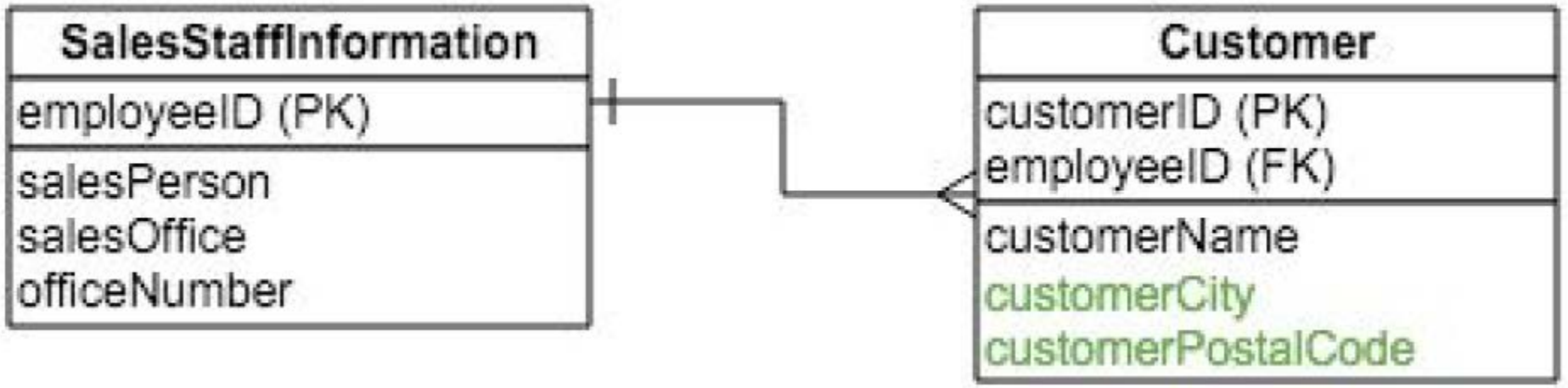
- Nemmere at søge
- Filtrere information
- Sortere information

Løsning for 1NF

SalesStaffInformation			
employeeID	salesPerson	salesOffice	officeNumber
1003	Mary Smith	Chicago	312-555-1212
1004	John Hunt	New York	212-555-1212
1005	Martin Hap	Chicago	312-555-1212

customer		
customerID	employeeID	customerName
1	1003	Ford
2	1003	GM
3	1004	HP
4	1004	Dell
5	1004	Apple
6	1005	Boeing

Løsning for 1NF



Løsning for 1NF

- Customer virker iht. 1NF
  - Nu kan vi sortere customers, hvilket var svært før
  - Opdatering af customers er også nemmere
  - At slette i customer tabellen var umuligt før
  - At tilføje kolonner til customers er nu muligt

1st normal form (1NF)

Definition

- En tabel er i sin første normal form (1NF) når og
- kun når:
  - Der er en primær nøgle der unikt identificerer hver række
  - Hver kolonne indeholder en atomar værdi og der ikke er gentaget indhold i kolonnen.

Formål

- Nemmere at søge
- Filtrere information
- Sortere information

2nd normal form (2NF)

Definition:

- En tabel er i sin anden normal form (2NF) når og kun når:
  - Tabellen er i 1. normal form og
  - Alle non-key kolonner er afhængige af tabellens primære nøgle.

Formål

- Der skal være en specifik nøgle der identificerer hver enkelt række.
  - En test kan laves:
    - Kan denne række beskrive hvad den primære nøgle identificerer.
      - Ja - rækken er afhængig af nøglen.
      - Nej - der skal oprettes en ny tabel.

Hvor er problemet med **SalesStaffInformation**

SalesStaffInformation			
employeeID	salesPerson	salesOffice	officeNumber
1003	Mary Smith	Chicago	312-555-1212
1004	John Hunt	New York	212-555-1212
1005	Martin Hap	Chicago	312-555-1212

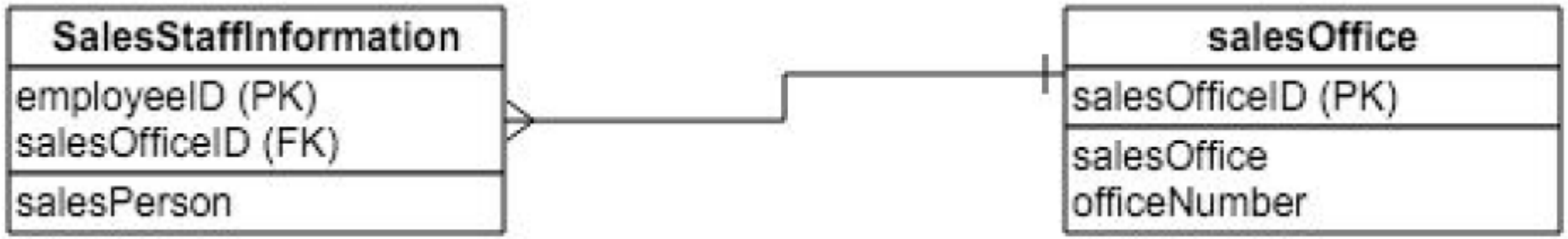
Her er problemet

SalesStaffInformation			
employeeID	salesPerson	salesOffice	officeNumber
1003	Mary Smith	Chicago	312-555-1212
1004	John Hunt	New York	212-555-1212
1005	Martin Hap	Chicago	312-555-1212

Derfor deler vi det på to tabeller

SalesStaffInformation		
employeeID	salesPerson	salesOfficeID
1003	Mary Smith	1
1004	John Hunt	2
1005	Martin Hap	1

SalesOffice		
salesOfficeID	salesOffice	officeNumber
1	Chicago	312-555-1212
2	New York	212-555-1212



Hvor er problemet med **customer**

customer				
customerID	employeeID	customerName	customerCity	customerPostalCode
1	1003	Ford	Dearborn	48123
2	1003	GM	Detroit	48123
3	1004	HP	Palo Alto	94303
4	1004	Dell	Austin	78720
5	1004	Apple	Cupertino	95014
6	1005	Boeing	Chicago	60601

Her er problemet

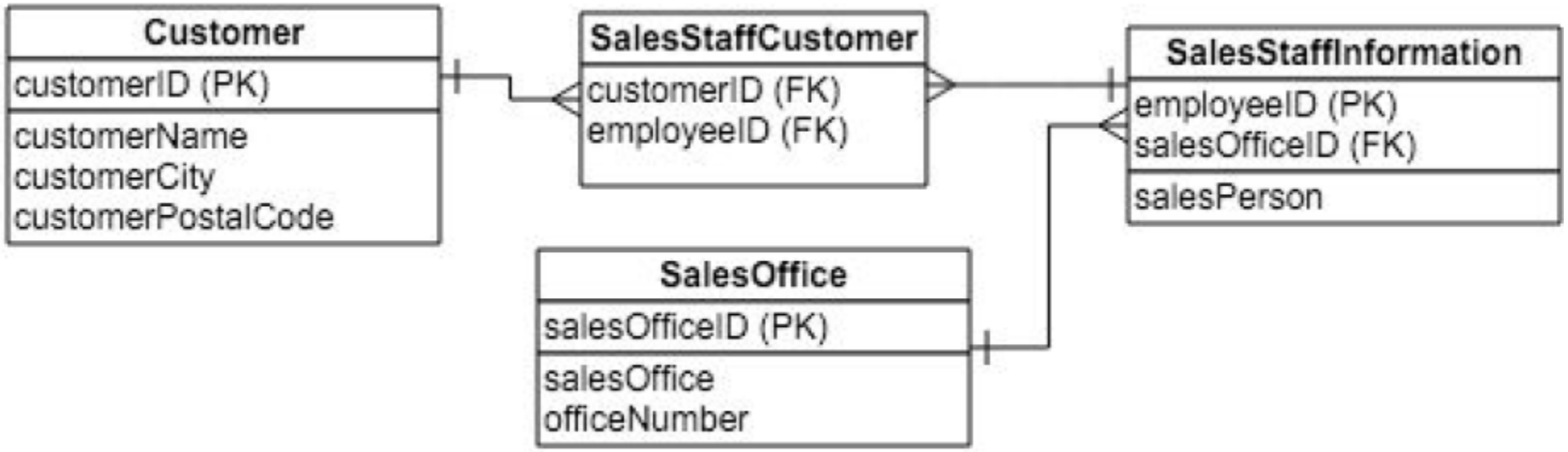
customer				
customerID	employeeID	customerName	customerCity	customerPostalCode
1	1003	Ford	Dearborn	48123
2	1003	GM	Detroit	48123
3	1004	HP	Palo Alto	94303
4	1004	Dell	Austin	78720
5	1004	Apple	Cupertino	95014
6	1005	Boeing	Chicago	60601

Igen deler vi det på to tabeller

customer			
customerID	customerName	customerCity	customerPostalCode
1	Ford	Dearborn	48123
2	GM	Detroit	48123
3	HP	Palo Alto	94303
4	Dell	Austin	78720
5	Apple	Cupertino	95014
6	Boeing	Chicago	60601

SalesStaffConsumer	
customerID	employeeID
1	Ford
2	GM
3	HP
4	Dell
5	Apple
6	Boeing

Et overblik over tabellerne



3. normal form (3NF)

Definition:

- En tabel er i sin tredje normal form (3NF) når og kun når:
  - Tabellen er i 2. normal form og
  - Der ikke findes felter uden for primærnøglen, som kun er afhængige af en del af primær-nøglen

Formål

- Der må ikke være indhold der er afhængig af noget andet end den primære nøgle. Det kunne være land og landets ID.
  - DK, Denmark
  - ES, Spanin
  - SE, Sweden
  - US, United States
- Værdierne ovenover kan ikke være i en tabel sammen med anden data.

Hvor er problemet med **Customer**

Customer			
customerID	customerName	customerCity	customerPostalCode
1	Ford	Dearborn	48123
2	GM	Detroit	48123
3	HP	Palo Alto	94303
4	Dell	Austin	78720
5	Apple	Cupertino	95014
6	Boeing	Chicago	60601

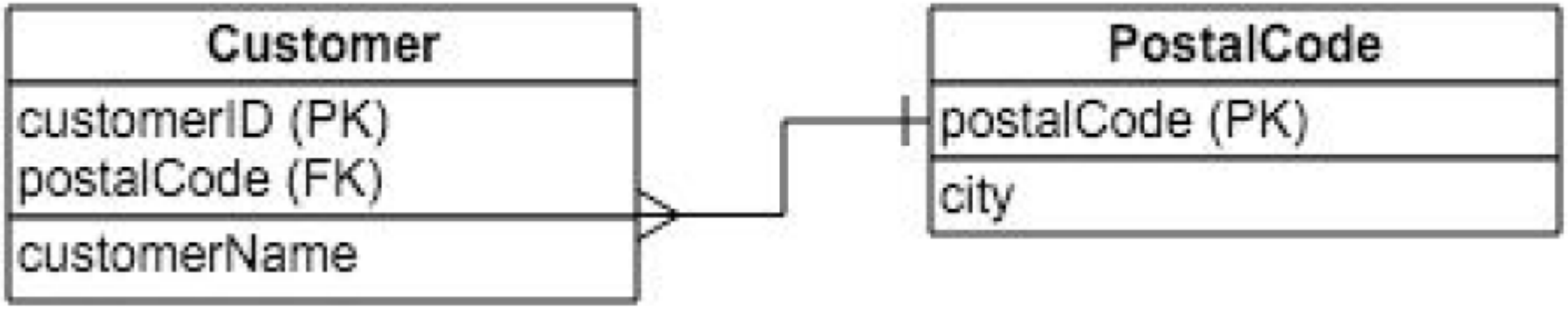
Her er problemet

Customer			
customerID	customerName	customerCity	customerPostalCode
1	Ford	Dearborn	48123
2	GM	Detroit	48123
3	HP	Palo Alto	94303
4	Dell	Austin	78720
5	Apple	Cupertino	95014
6	Boeing	Chicago	60601

Også her deler vi det på to tabeller

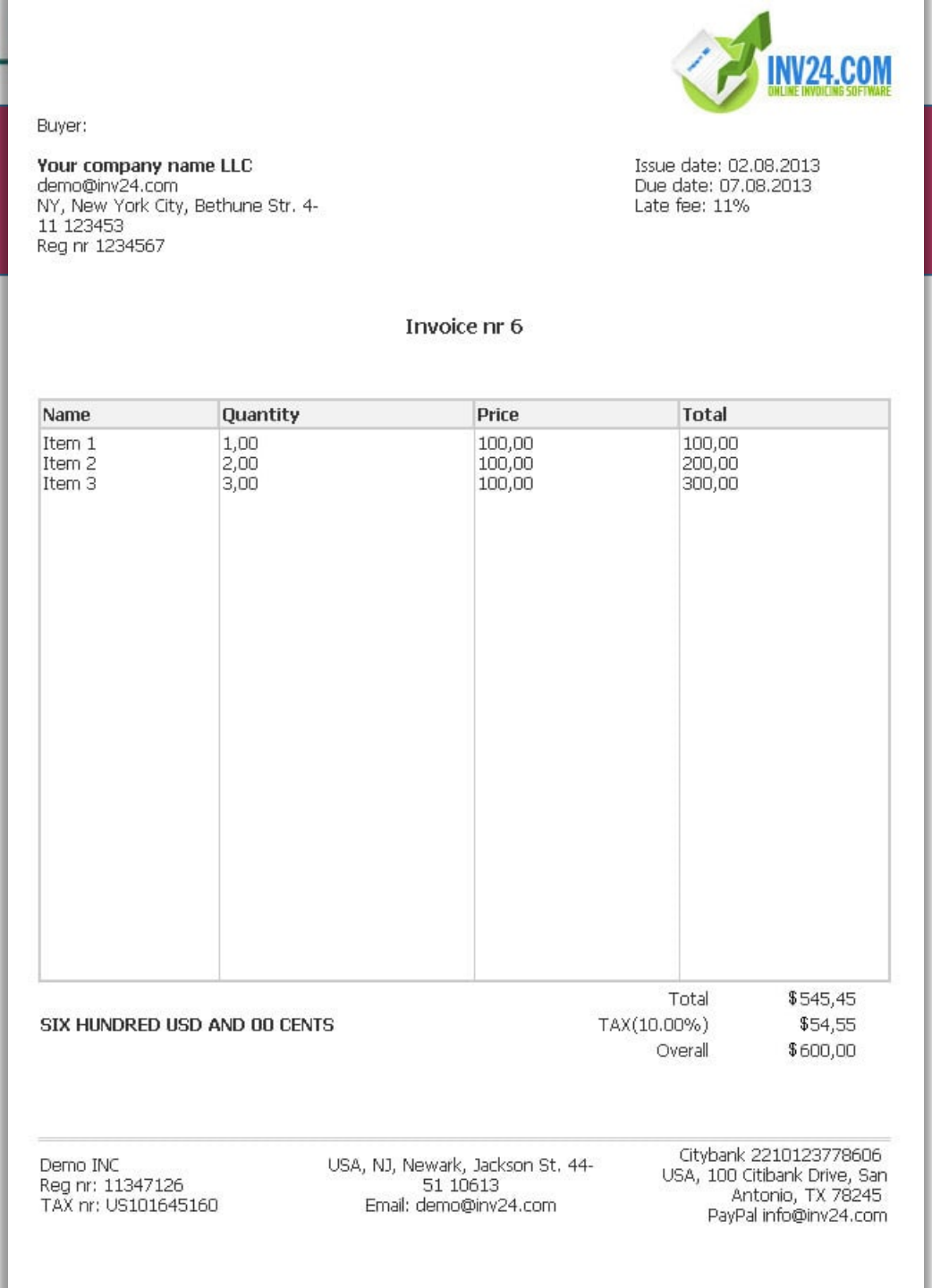
Customer	
customerID	customerName
1	Ford
2	GM
3	HP
4	Dell
5	Apple
6	Boeing

PostalCode	
postalCode	city
48123	Dearborn
48123	Detroit
94303	Palo Alto
78720	Austin
95014	Cupertino
60601	Chicago



Opgave

- Opbyg strukturen til en database de tre normal former for fakturaen



INV24.COM
ONLINE INVOICING SOFTWARE

Buyer:

Your company name LLC
demo@inv24.com
NY, New York City, Bethune Str. 4-
11 123453
Reg nr 1234567

Issue date: 02.08.2013
Due date: 07.08.2013
Late fee: 11%

Invoice nr 6

Name	Quantity	Price	Total
Item 1	1,00	100,00	100,00
Item 2	2,00	100,00	200,00
Item 3	3,00	100,00	300,00

	Total	$545,45
SIX HUNDRED USD AND 00 CENTS	TAX(10.00%)	$54,55
	Overall	$600,00

Demo INC
Reg nr: 11347126
TAX nr: US101645160

USA, NJ, Newark, Jackson St. 44-
51 10613
Email: demo@inv24.com

Citybank 2210123778606
USA, 100 Citibank Drive, San
Antonio, TX 78245
PayPal info@inv24.com

Introduktion til MySQL Workbench

System til alle former for arbejde med MySQL databaser

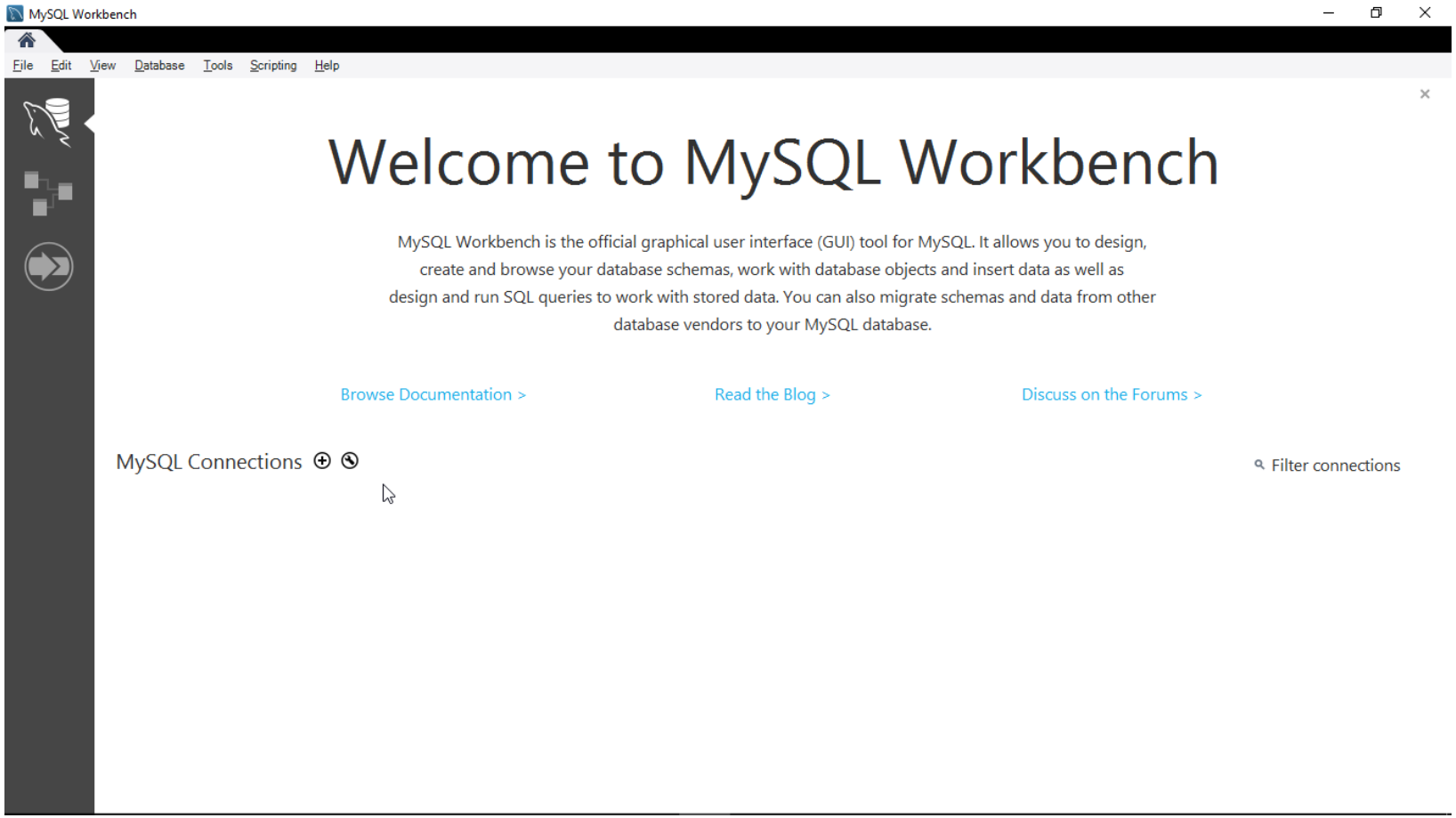
Hvad er MySQL Workbench

- MySQL Workbench er et grafisk værktøj til at arbejde med MySQL-servere og databaser.
- MySQL Workbench-funktionalitet dækker blandt andet:
  - **SQL-udvikling**: Gør det muligt at oprette og administrere forbindelser til databaseservere. Sammen med at du kan konfigurere forbindelsesparametre, giver MySQL Workbench evnen til at udføre SQL-forespørgsler på databaseforbindelserne ved hjælp af den indbyggede SQL Editor.
  - **Data Modeling** (Design): Gør det muligt at oprette modeller af dit databaseskema grafisk, omvendt og fremadrettet mellem et skema og en live database og redigere alle aspekter af din database ved hjælp af den omfattende tabel editor. Tabeleditoren indeholder nemme at bruge faciliteter til redigering af tabeller, kolonner, indekser, udløsere, partitionering, valgmuligheder, indsætninger og privilegier, rutiner og visninger.
  - og mere til…

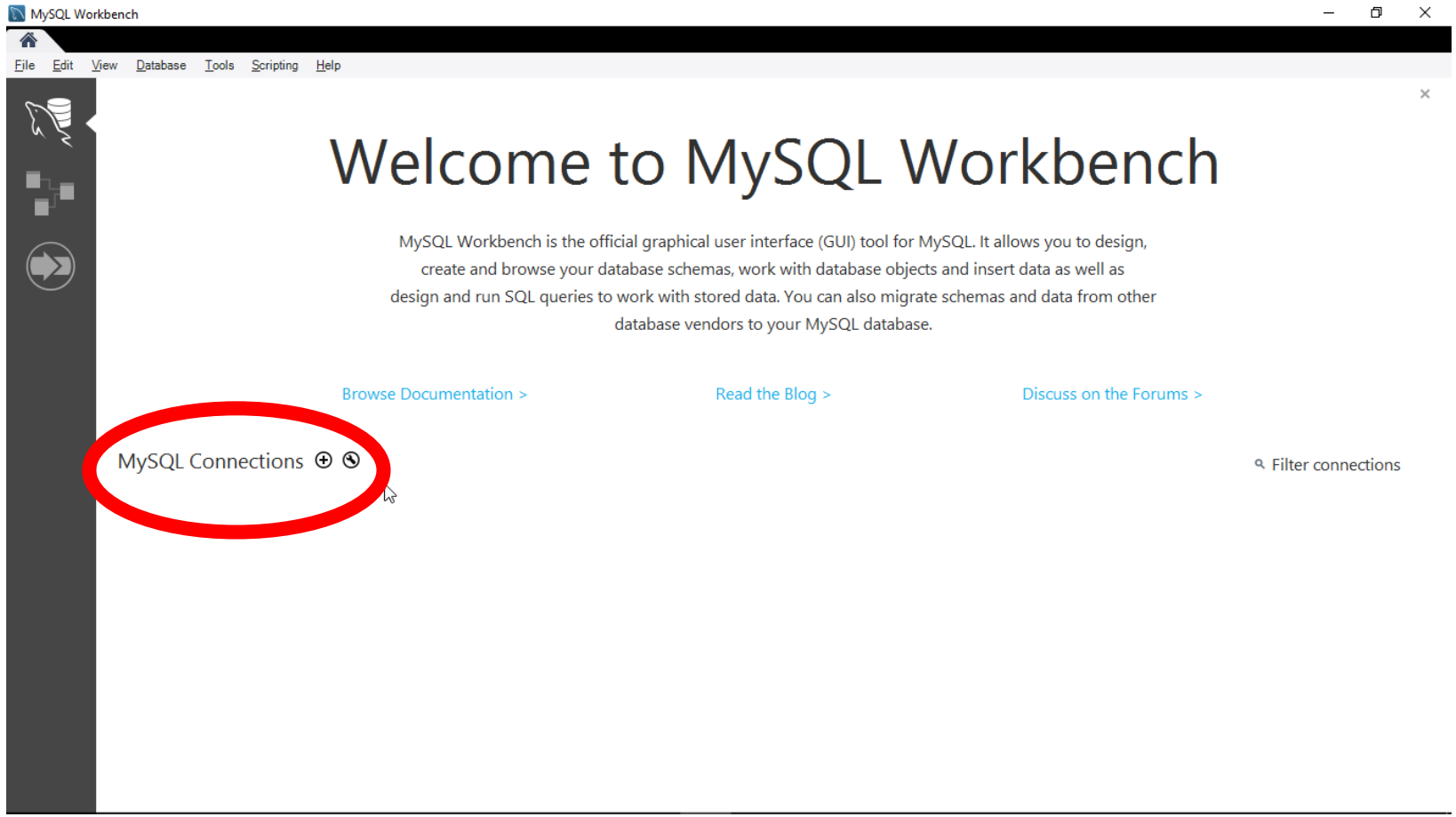
Installer MySQL Workbench

- Hent det gratis program fra https://dev.mysql.com/downloads/workbench/
- Installer det

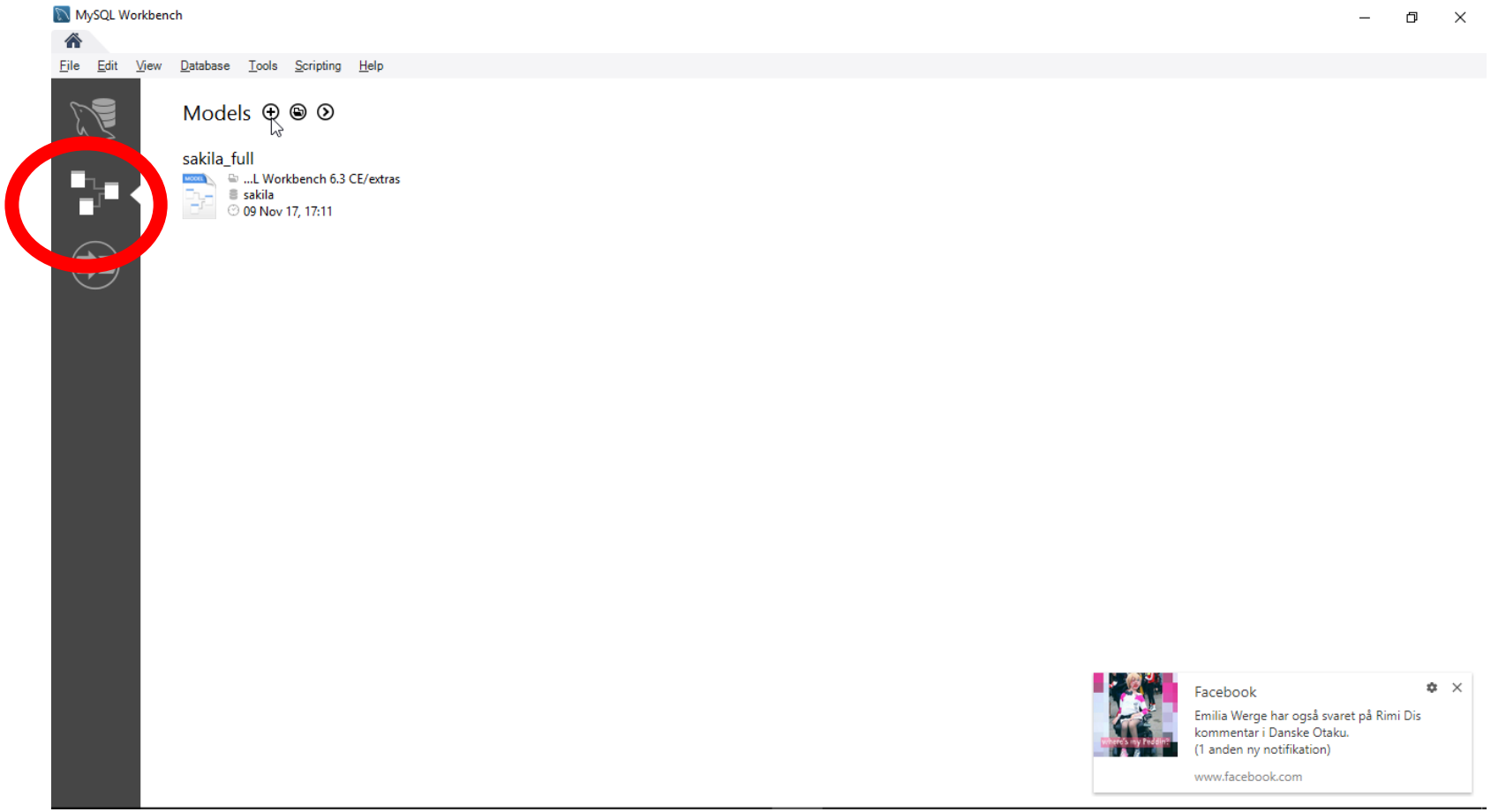
Start MySQL Workbench



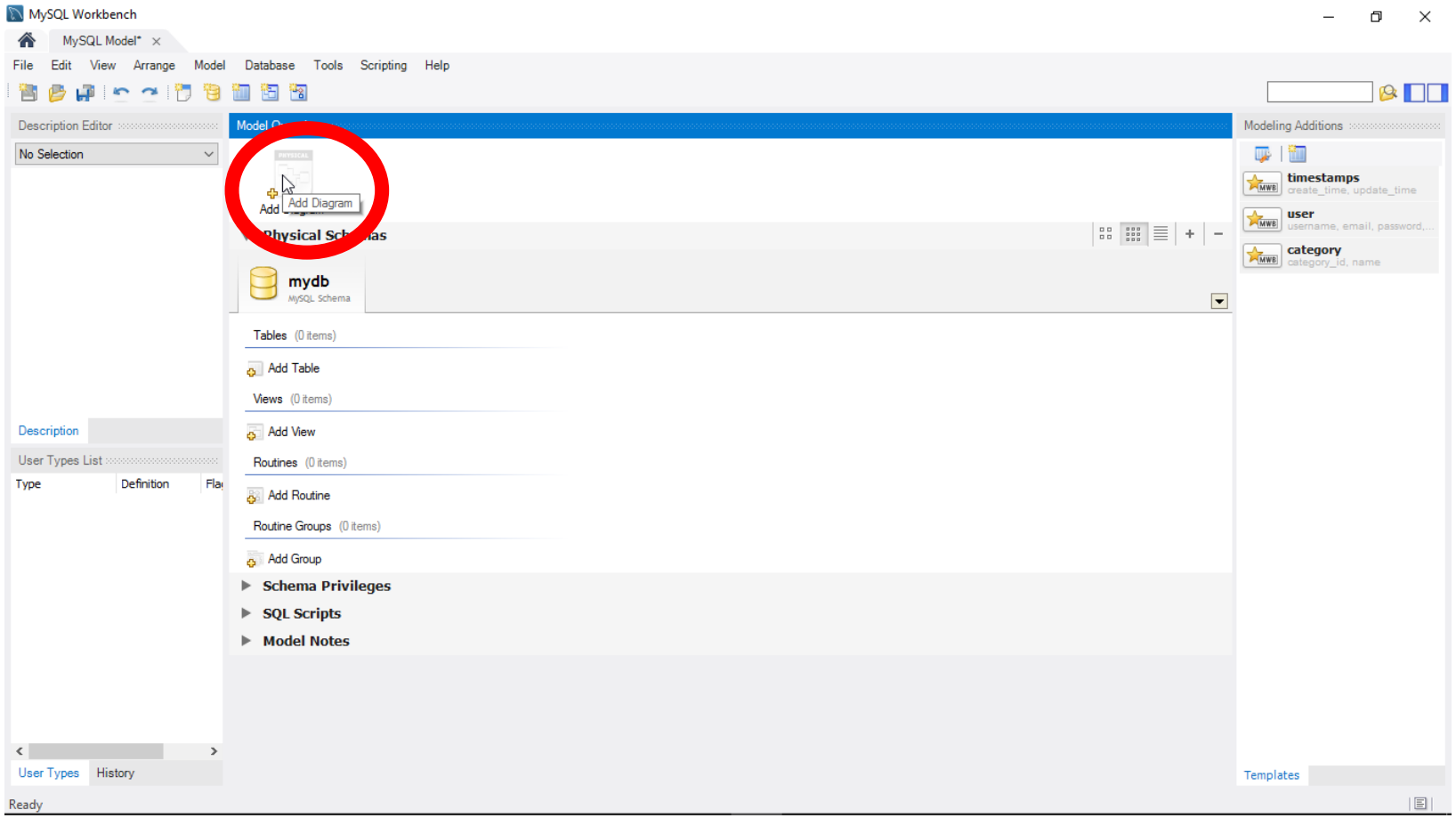
Her opretter du forbindelse til ekserne databaser du vil styre – det har vi umiddelbart ikke brug for



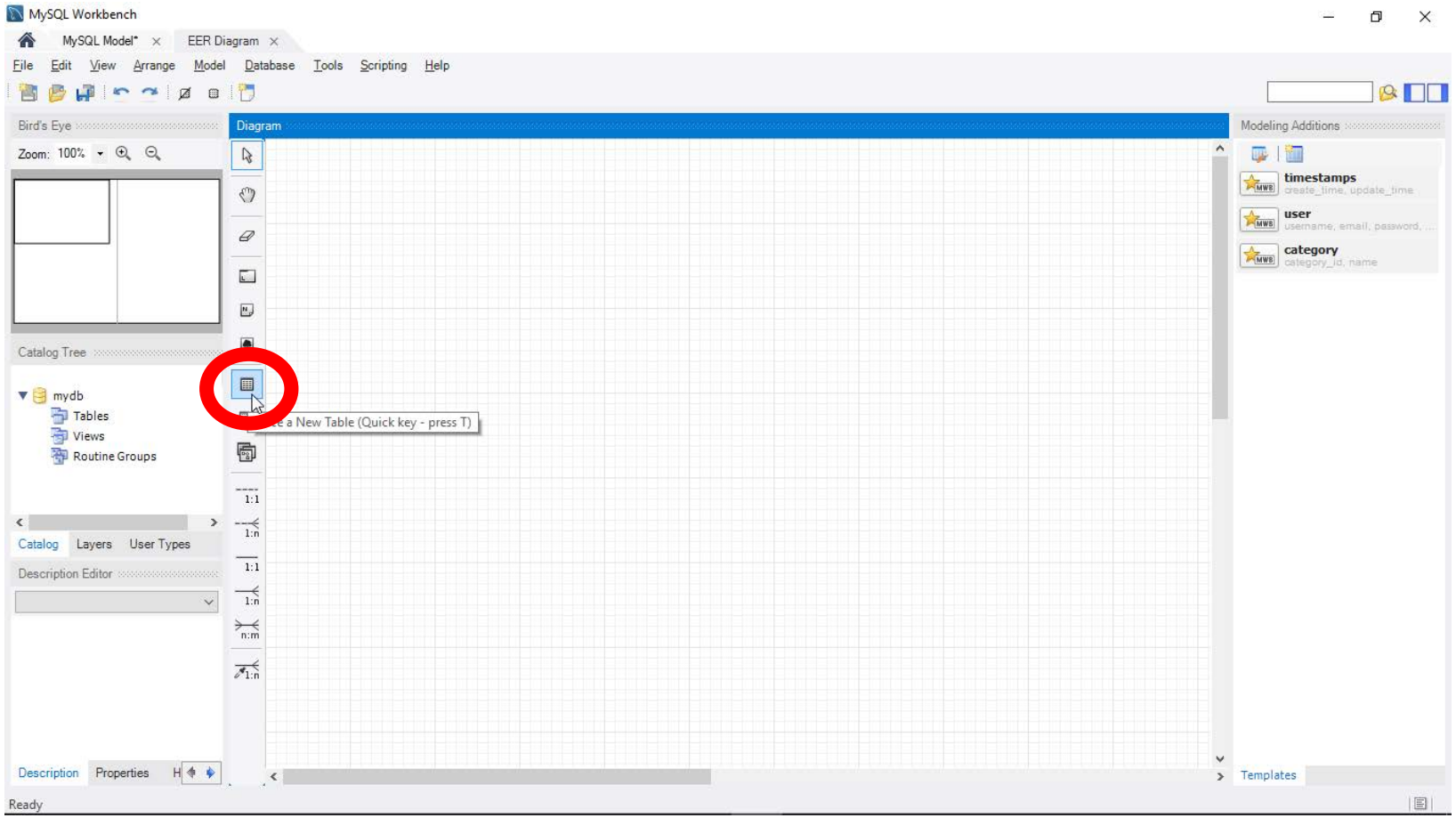
Her laver man ny modeller og diagrammer



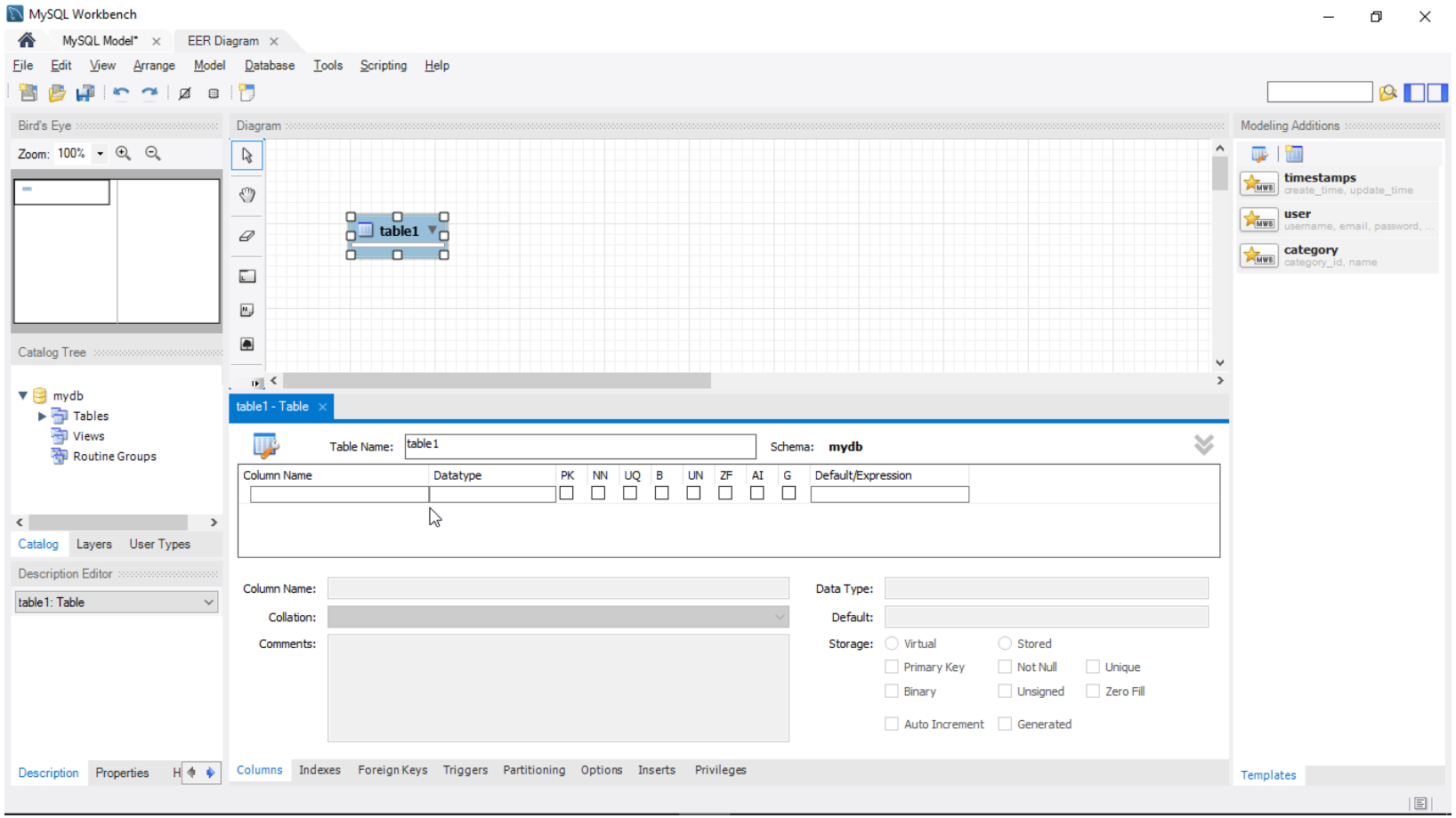
Her kan du lave ny ER diagrammer



Her laver du en ny tabel



Når du har lavet en tabel dobbeltklikker du på den for at redigere den



Lav et diagram

- Det skal omhandle det område du sagde du ville organisere i starten af timen
- Det skal indtil videre indeholder tre tabeller med minimum ni kolonner samlet
  - Du behøves ikke tænke på fremmede nøgler, dem gennemgår vi næste gang
- Vælg passende datatyper
- Vi føjer flere tabeller til næste gang

Opsummering af i dag

Hvad vi har gennemgået i dagens lektion

Lektier til næste gang

Læsning

Læsning til næste gang

- Læs side 7 – 53 i SQL - The Ultimate Beginners Guide

Kilder

Materiale benyttet i denne undervisningsgang

Kilder
Konceptet bag databaser

- https://www.slideshare.net/Uniqueangel1/different-type-of-databases
- http://www.jegsworks.com/lessons/ComputerBasics/lesson1-2/lesson2-4database.htm

Flad fil databaser

- https://www.techopedia.com/definition/25956/flat-file
- https://en.wikipedia.org/wiki/Flat_file_database
- https://en.wikipedia.org/wiki/INI_file
- https://en.wikipedia.org/wiki/Comma-separated_values

Kilder
Konceptet bag databaser

Hierarkiske databaser

- https://simple.wikipedia.org/wiki/Hierarchical_database_model
- https://mariadb.com/kb/en/library/understanding-the-hierarchical-database-model/
- https://www.techopedia.com/definition/19782/hierarchical-database
- http://databasemanagement.wikia.com/wiki/Category:Hierarchical_Data_Model
- http://www.ukessays.co.uk/essays/information-technology/hierarchical-data-model.php
- https://youtu.be/1Wnt1XIYMoM

Kilder
CRUD

- https://stackify.com/what-are-crud-operations/
- http://sqlfiddle.com/
- http://www.tutorialspoint.com/mysql/mysql-insert-query.htm
- http://www.tutorialspoint.com/mysql/mysql-update-query.htm
- http://www.tutorialspoint.com/mysql/mysql-update-query.htm
- http://www.tutorialspoint.com/mysql/mysql-delete-query.htm

Kilder
ER diagrammer

- https://www.lucidchart.com/pages/ER-diagram-symbols-and-meaning
- http://en.wikipedia.org/wiki/Entity–relationship_model