Ett arbetsexempel – Faktureringsrutin

Detta dokument är skrivet för att i första hand förstå den process som äger rum och vilka steg som man ska genomföra och att förstå vad som utförs i de tre viktiga stegen Konceptuell, Logisk och Fysisk modell.

Ett företag använder idag en faktureringsrutin där man fakturerar med hjälp av en ordbehandlare och man har insett att det inte håller i längden då man vill ha bättre kontroll på vad man fakturerat och man vill ha statistik mm.

Skriv ut fakturan så du har den framför dig när du läser detta.

Det första man kan fundera över är vilka personer som ska vara med i utveckling av faktureringssystemet. Man ser av fakturan att det kommer att beröra ett flertal personer i företaget.

- Ekonomiavdelning, som fakturerar och sköter ekonomisk uppföljning
- Lageravdelningen, artiklar kommer att finnas på lagret
- Försäljningsavdelning – de säljer ju artiklar
- Marknadsföring som har kontakt med kunderna

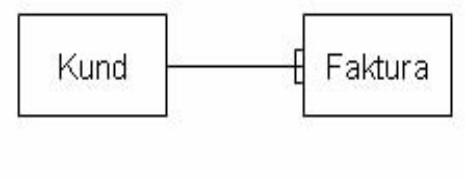
Det är viktigt att förstå att många personer blir inblandade i en process där uppgiften är att utveckla ett nytt system med en ny databas och dessa ska stödja verksamheten i företaget. Se i Livscykelmodellen och studera raden Deltagare i diskussionen.

För att uppnå ett lyckat resultat bör man se till att berörda personer får delta i utvecklingen. Det får inte bli så att någon säger: ”Jag fick inte vara med så jag bryr mig inte om att använda det nya systemet.”

Den Konceptuella modellen

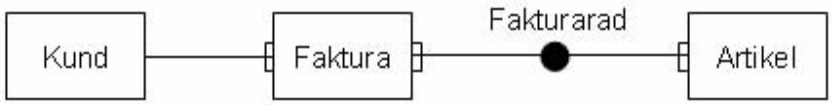
Ett första steg i datamodelleringen är att försöka finna de objekt som är viktiga för verksamheten. När man funnit dessa så har man de första viktiga objekten till databasen. Studerar du fakturan kommer du troligtvis att finna tre objekt. Kund, Faktura och Artikel. När du namnger dessa objekt ska du ange namn i singularis. Undvik gärna ö, ä och ö när du namnsätter objekten då de ibland är svåra att hantera senare i utvecklingsprocessen. När vi utvecklar den konceptuella modellen så ska vi inte ha någon databas i åtanke. Det kommer senare i utvecklingen.

Ange hur relationen är mellan Kund och Faktura. Tänk på en kund och hur många fakturor en kund kan ha under en längre tid. Du bör då få att en kund kan ha flera fakturor. Omvänt kan en faktura endast ha en kund. Det betyder att du får en modell som ser ut på följande sätt:



Mellan Faktura och Artikel gäller följande:
En Faktura kan innehålla många artiklar. En Artikel kan finnas på många fakturor. Vi får då en n:m relation, dvs många till många relation. Följande modell har vi fått.

Det som inträffar när man finner en n:m relation är att vi får ett sk relationsobjekt . Relationsobjektet sätts i vårt fall in som en punkt mellan Faktura och Artikel. Relationsbjektet är de rader som ingår i fakturan och innehåller alltså de artiklar som vi säljer. Objektet Artikel är alla de artiklar som vi kan sälja. En viss kund köper dock bara några få av dem.



Nu kan man naturligtvis gå vidare och utreda hur det ser ut med Artikel och varifrån man köper dessa och finner då Leverantör och kanske hitta ytterligare objekt som är viktiga. I detta enkla utbildningssyfte stannar vi enligt modellen ovan och börjar titta på de datamängderna som ingår i respektive objekt. Det kommer senare att ge vilka fält som ska ingå i respektive objekt som senare blir tabeller i databasen.

För **kund** tänker vi oss följande datamängd:

Kundid	som identifierare på resp kund. Unikt värde, Pk.
Namn	namn, 40 tecken
Adress	gatuadressen, 30 tecken
Postnr	postnumret, omfattas av 6 tecken med blanksteg, utan blanksteg 5 tecken
Ort	ortsnamnet, 30 tecken
Tel	telefonnummer till kunden
Mobil	mobilnumret till kunden

Faktura

Faktid	som identifierare på resp faktura, fakturanr. Unikt värde, Pk.
Kundid	som koppling mot kund – vilken kund har denna faktura, Fk
Datum	När köpet inträffar, SSÅÅ-MM-DD, datumformat, 10 tecken
Betvillkor	antal dagar innan faktura förfaller, 2 siffrig tal

Artikel

Artikelid	som identifierare på resp artikel. Unikt värde, Pk.
Artnamn	artikelnamnet, 30 tecken
Antal	som finns i lager, 4-siffrigt tal utan decimaler
Pris	pris per enhet, 6-siffrigt tal med 2 decimaler
Enhet	enhet, st, lpm, kg eller liknande
Moms	moms för artikeln, 4-siffrigt tal med 2 decimaler

Fakturarad

Faktid	som koppling mot faktura - vilken faktura raden tillhör
Artikelid	som koppling mot artikel – vilken artikel ligger på denna rad
Artnamn	artikelnamnet, 30 tecken
Antal	som man säljer, 4-siffrigt tal utan 2 decimaler
Pris	pris per enhet, 6-siffrigt tal med 2 decimaler
Enhet	enhet, st, lpm, kg eller liknande
Moms	moms för artikeln, 4-siffrigt tal med 2 decimaler

Fälten Faktid och Artikelid bildar en sammansatt nyckel, Pk.

Faktura behöver inte ha det uträknade beloppet, nettovärde, momsbelopp då detta kan härledas vid varje tillfälle. På samma sätt gäller förfallodatum. Det härleds ur Datum och Betvillkor.

Fakturarad behöver egentligen inte innehålla Artikelnamn och pris då dessa finns i tabellen Artikel. Men då uppstår problemet om man köper en artikel vid ett visst tillfälle så visas ett belopp och ett namn. Döper man sedan om artikeln eller ger artikeln ett annat pris så kommer fakturan att påverkas efter försäljningen. De redan sålda artiklarna som är ändrade kommer alltså att förändra de gamla fakturorna.

Innan man fortsätter med arbetet att utveckla datamodellen så bör man fundera på vad man redan har gjort och om det är rätt. Det kanske är så att man ska lägga till flera objekt eller kanske förändra de befintliga. Detta är en utvecklingsprocesss och den bör få ta tid. I denna process så kanske man kommer fram till att vi borde ha med kontaktpersoner hos kund. Vi kanske ska ha rabattsatser på artiklar och varje kund ska ha en rabattsats beroende på hur mycket man handlar och varje faktura ska ha en egen rabattsats beroende på fakturans ekonomiska omfång. Det kan alltså tillkomma ett antal objekt när man får tid att fundera. Det kanske också är också lämpligt att försöka få in flera objekt för att ”utmana” situationen.

För att den konceptuella modellen ska bli färdig ska man rita tabellerna med data i för att se bättre om man tänkt rätt.

Den Logiska modellen

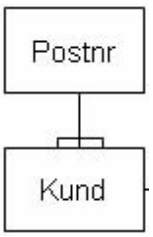
Nästa steg är att skapa den logiska modellen som utgår från den konceptuella modellen. I utveckling av den konceptuella modellen ska man heller inte ha en slutlig databas i åtanke. Den kommer in i det tredje och sista steget när vi utvecklar den fysiska modellen. Vi vet alltså ännu inte vilka egenskaper vår databas har.

Ett mycket viktigt steg vi tar nu är att genomföra normalisering. Syftet med detta är att bland annat hitta fält som ligger i fel tabell, dubbellagring av data (redundanta data)och lite annat. Vi använder oss av fyra normalformer:

1 NF (normalformen)
Unik nyckel och icke delbara fält.
Gå genom och kontrollera alla objekten, tabellerna, att de har unika nycklar och icke delbara fält. Se mer i boken om detta.

2NF
Om tabellen är på 1NF och dessutom Alla icke nyckelfält ska var funktionellt beroende av hela nyckeln. Denna kommer speciellt ifråga när man har sammansatt nyckel som vi har i Fakturarad. Kontrollera att alla fält är beroende av hela nyckeln. Se mer i boken om detta.

3NF
Om tabellen är på 2NF och dessutom Det får inte finnas några funktionella beroende mellan icke nyckelfält. I Kund så hittar vi exempelvis Ort är beroende av Postnr. Vi bryter därför isär Kund och skapar två objekt. Kund och Postnr: Se mer i boken om detta funktionella beroende mellan icke nyckelfält.



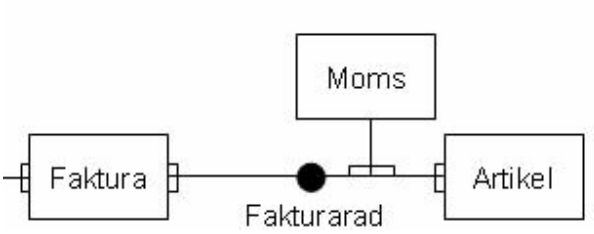
Det innebär förändringar i Kundobjektet: För **kund** tänker vi oss följande datamängd:

Kundid	som identifierare på resp kund. Unikt värde, Pk.
Namn	namn, 40 tecken
Adress	gatuadressen, 30 tecken
Postnr	postnumret, omfattas av 6 tecken med blanksteg, utan blanksteg 5 tecken
Tel	telefonnummer till kunden
Mobil	mobilnumret till kunden

Postnr

Postnr	postnumret, omfattas av 6 tecken med blanksteg, utan blanksteg 5 tecken, Pk, Unik
Ort	ortsnamnet, 30 tecken

I Artikel så hittar vi moms som ett fält som egentligen inte är beroende av artikel. Den kommer att vara med i försäljningstillfället och bör därför ligga ihop med Fakturarad. Moms blir med andra ord beroende av Fakturarad. Vi skapar ett nytt objekt som innehåller moms.



Det innebär att vi tar bort Moms ur Artikel och skapar en ny tabell för Moms.

Artikel

Artikelid	som identifierare på resp artikel. Unikt värde, Pk.
Artnamn	artikelnamnet, 30 tecken
Antal	som finns i lager, 4-siffrigt tal utan decimaler
Pris	pris per enhet, 6-siffrigt tal med 2 decimaler
Enhet	enhet, st, lpm, kg eller liknande

Moms

Momsid	som identifierare på resp momstyp. Unikt värde, Pk.
Moms	moms för artikeln, 4-siffrigt tal med 2 decimaler

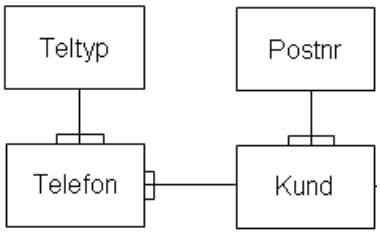
Vi genomför ingen förändring i Fakturarad eftersom vi vill lagra den moms som gäller för det aktuella tillfället.

4NF

Om tabellen är på 3NF och dessutom Ett attribut får endast finns en gång per tabell

I Kund så finns fälten Tel och Mobil och de innehåller egentligen samma uppgifter. Antag att en Kund vill ha ytterligare en telefon då måste vi bygga om applikationen eftersom det tillkommer ett nytt fält.

Vi får ett nytt objekt som heter Telefon: Eftersom vi också är beroende av att veta vad respektive telefonnr används till, hem, arbete, mobil etc läger vi in ett objekt för detta. Vi kallar det nya objektet Teltyp.



En kund kan ha flera telefonnummer men ett nummer går bara till en kund.

Följande förändringar genomförs:
För **Kund** tänker vi oss följande datamängd:

Kundid	som identifierare på resp kund. Unikt värde, Pk.
Namn	namn, 40 tecken
Adress	gatuadressen, 30 tecken
Postnr	postnumret, omfattas av 6 tecken med blanksteg, utan blanksteg 5 tecken

Telefon

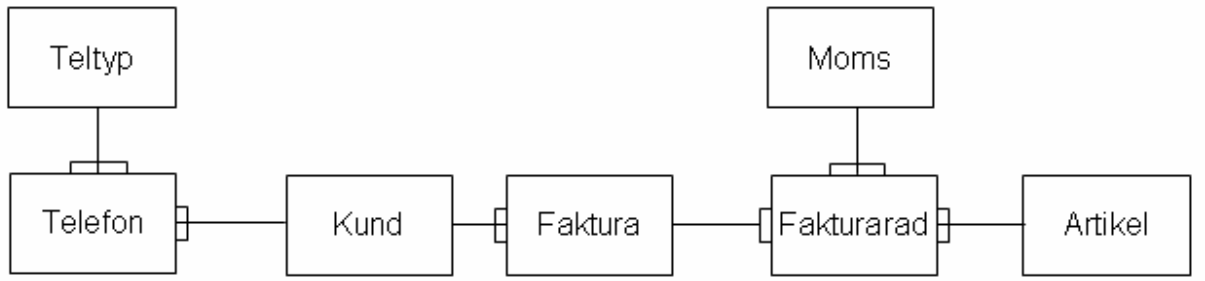
Telid	som identifierare på resp telefonrad. Unikt värde, Pk.
Telenr	telefonnummer inkl lands o riktnr, 20 tecken
Kundid	koppling mot kund, Fk
Teltypid	Koppling mot typtabellen, Fk

Teltyp

Teltypid	som identifierare på resp teletyp. Unikt värde, Pk.
Teltyp	vad det är för typ anges, 10 tecken

Om du nu har gått genom alla tabellerna och kontrollerat alla normalformerna så är hela normaliseringsprocessen klar. Processen skapar ett antal fler objekt än vad man hade från den konceptuella modellen.

Genomför nu också objektifiering av de relationsobjekt som finns i datamodellen. Vi har ett relationsobjekt, Fakturarad, och när vi objektifierar detta så bygger vi ett ”riktigt” objekt av relationsobjektet. Det innebär dels att vi ritar om modellen och ritar en rektangel istället för en punkt. Vi vänder också på relationssymbolerna, gafflarna:



Vi för en diskussion om hur det fungerar med nycklar i fakturarad. Antag följande:
Kalle ringer och vill köpa fyra paket CD-skivor till sig och sedan vill han köpa två paket DVD-skivor. Men så vill han köpa till sin granne ett paket CD-skivor – samma som han köpte till sig själv men han vill ha dem på en separat rad så man lätt ser vad de kostar eftersom hans granne ska betala detta. I vårt system går inte detta eftersom Pk är sammansatt av Faktid och Artikelid.

Detta löser vi genom att vi lägger till ett fält till i Fakturarad. Fältet blir en egen räknare som blir en unik identifierare. Detta tillsammans med att man ritar relationsobjektet som en rektangel kallas för objektifieringen. Följande tabell får vi för Fakturarad:

Fakturarad
Fakturaradid som identifierare på fakturarad. Unikt värde, Pk.
Faktid som identifierare vilken faktura raden tillhör, Fk
Artikelid som koppling mot artikel – vilken artikel ligger på denna rad, Fk
Artnamn artikelnamnet, 30 tecken
Antal som man säljer, 4-siffrigt tal utan 2 decimaler
Pris per enhet, 6-siffrigt tal med 2 decimaler
Enhet st, lpm, kg eller liknande
Moms för artikeln, 4-siffrigt tal med 2 decimaler

Huruvida man ska sätta Fakturaradid, Faktid och Artikelid som en sammansatt nyckel eller inte beror på hur man resonerar om det är obligatoriskt att lägga in raden på en faktura, dvs måste faktid finnas med och om det är obligatoriskt med en artikel och då måste det finnas med ett artikelid. Som en tankeställare kan man säga att det kan vara bra om man kan skriva en egen text på en rad som inte tillhör någon artikel.

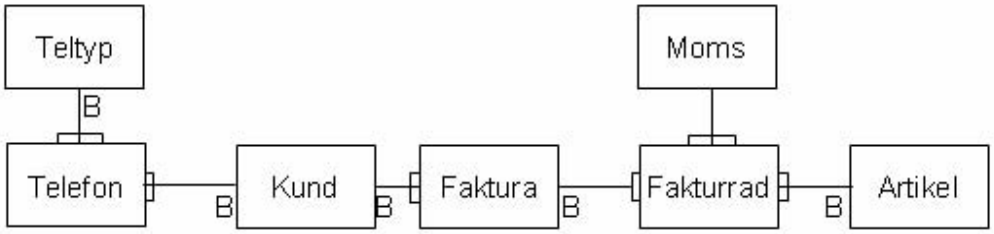
Den Fysiska modellen

Nu ska vi ha en databashanterare i åtanke så att vi kan bestämma hur fälttyper ska vara och vi kan ha andra saker med oss avseende hur den specifika databashanteraren fungerar och dess egenskaper.

Vi genomför nu denormaliseringar. Det innebär exempelvis att vi säger att om vi har kvar Postnr tabellen så är vi tvungna att koppla samman två tabeller för att få ett komplett svar på adressen som en viss Kund har. Vi ser inte det som något större merarbete att låta Ort ligga i Kund och att manuellt skriva in ortsnamnet på nytt för varje ny Kund. Om postverket byter postnr på ett ortsnamn kan vi använda Update (SQL) för att byta ortsnamnet för alla poster som har ett visst postnr eller tvärtom. Visserligen får vi en mängd med dubbletter. Resultatet av detta ger att vi slår ihop Postnr med Kund igen.

Det är viktigt att också försöka se vilka generaliseringar och optimeringar som vi kan göra i form av kolumnvisa sammanslagningar och kanske kolumnvisa och radvisa delningar. De sistnämnda kanske främst kommer ifråga om vi ska ha tillgång till historik. För att inte tynga ner en del tabeller kanske man ska kunna flytta över data till historik efter de varit aktiva i exempelvis två år. Därefter tillhör de historiken

Den slutliga modellen ser nu ut på följande sätt:



B som du hittar vid vissa objekt anger att det finns ett Beroende. Det innebär exempelvis att en post i telefon inte ska kunna läggas upp utan att det finns en post i Kund. Telefon är beroende (obligatoriskt) av Kund. Dvs vi måste lägga in Kundid i telefontabellen.

För att den fysiska modellen ska bli helt klar ska även tabellprecisering utföras så att alla ingående fält, fältnamn, fälttyper och fältlängder är angivna.