*Namn: Marco Terán Gutmanowitz*

*E-mail: marcogbg@hotmail.com*

*Datum: 2022-03-22*

*Kurskod: DDS200 – Databaser och Databasdesign*

**Titel: Databasmodellering & Design - Från idé till Databas**

**Caserapport av: Marco Terán Gutmanowitz**

**Innehåll**

[1 Inledning, bakgrund 3](#_Toc32997628)

[2 Modellering 3](#_Toc32997629)

[3 Implementering 3](#_Toc32997630)

[4 Transaktionshantering 3](#_Toc32997631)

[5 Distribuerade databassytem och Data Warehouse 3](#_Toc32997632)

[6 Big Data Ethics 3](#_Toc32997633)

[7 Referenser 4](#_Toc32997634)

[8 Bilagor 5](#_Toc32997635)

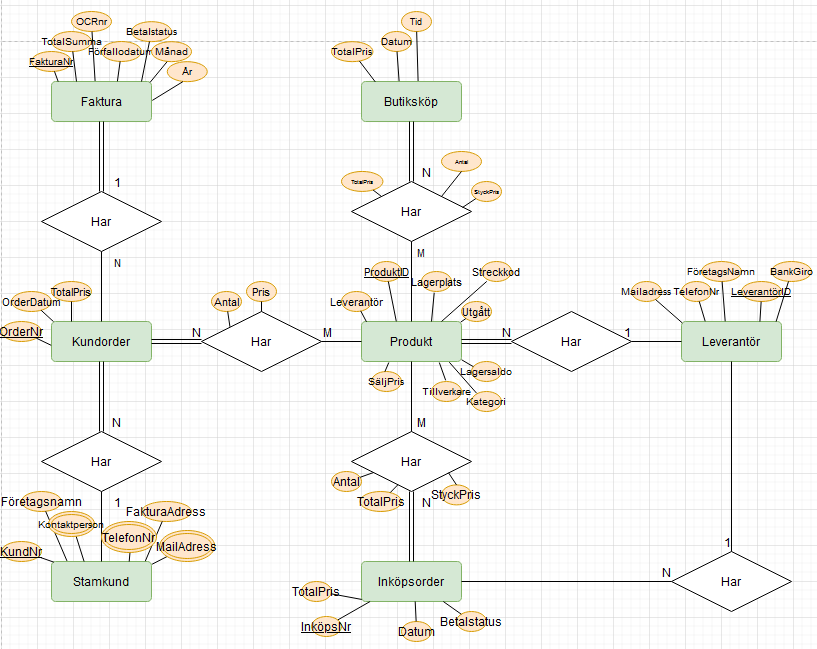
# Inledning, bakgrund

Denna rapport syftar till att på ett klartydligt språk dokumentera modelleringsprocessen och realiseringen av en relationsdatabas utifrån en idé eller ett önskemål utifrån ett kundperspektiv. I rapporten kommer jag att redogöra steg för steg vilka verktyg och principer som använts för just detta syfte.

Byggvaruhus AB är ett familjeföretag på 5 personer som är I behov av ett digitalt Order-Lager-Faktureringssystem (OLF-system) för sin mindre bygghandel de driver. Idag är bygghandelns stora problem lagerhanteringen, som utförs med pärm. Då mycket sker manuellt via pärmen så blir det ofta fel lagersaldon och man hinner exempelvis inte beställa in varor I tid utan att saldon på vissa varor når noll. Även order- och fakturahantering från kunder och leverantörer sker manuellt. Bygghandelns huvudaktörer består av en ekonomiansvarig, en kassör och 3 lagerarbetare.

‘’Idag har vi manuellt system som rör inköp, försäljning och lagerhantering. Ordrar sparar vi i en pärm, likaså fakturor vi skickar till våra kunder. Kontaktuppgifter till våra leverantörer har vi i ett excelark. Lagret är vårt stora bekymmer. Vi har problem med det manuella systemet. Det är bökigt när vi inte vet vad vi har på lagret. Det tar tid att inventera, händer ofta att vi är utan artiklar när de behövs. Leverantörer hade varit önskvärt att ha i systemet, för att kunna få bättre översikt över vilka produkter vi köper från respektive leverantör, och våra beställningar till leverantörerna. Det räcker om vi har leverantörer i systemet, våra inköpsordrar och även att vi kan i systemet markera att vi betalat leverantörsfaktura.’’ - Ekonomiansvarig, Byggvaruhus AB.

# Modellering

**Figur 1 – Tidigt ER-diagram för Byggvaruhus AB.**

Ett första utkast till ett Entity-Relationship Diagram (ER-Diagram) visas ovan för Byggvaruhus AB. Ett ER-Diagram är en typ av *konceptuell datamodell* som används för att skapa en tydlig modell över det som sedan skall matas in i en databashanterare (Databasteknik, 2005).

I grund och botten antar vi att verksamheten önskar att OLF-systemet skall hantera:

* Kundordrar
* Inköpsordrar
* Inköpsfakturor (Inköpsorder i Figur 1)
* Kundfakturor
* Produkter
* Stamkunder
* Leverantörer
* Köp i butik

Tanken bakom diagrammet är att gruppera in förutspådda data i grupper som senare skall bli tabeller, därför har vi alla ovannämnda punkter som entiteter med sina respektive attribut i ER-Diagrammet.

Entiteterna har sina respektive samband:

* Kundorder har ett starkt samband till Stamkund. En instans av en kundorder måste ha en stamkund för att kunna skapas. Den får inte vara ett s.k null-värde.
* En Fakturainstans måste på samma vis ha en kundorder för att kunna skapas. En Faktura kan inte skickas till en kund om inte en order har skapats.
* Samma logik gäller för Kundorder-Produkt, Butiksköp-Produkt, Inköpsorder-Produkt samt Produkt-Leverantör.

För entiteten Produkt har vi attributen:

* ProduktID (Primärnyckel) - För att kunna unikt identifiera varje produkt i tabellen.
* Lagerplats - För att kunna hitta varje produkts lagerplats ifrån OLF-systemet. Varje värde i kolumnen är unikt för att undvika dålig lagerhantering.
* Lagersaldo - För att kunna implementera eventuella varningssystem eller automatiska ordrar till leverantörer för lågt saldo för en viss produkt. Bra att ha.
* Utgått - Ett attribut som är bra att ha ifall man inte längre kan beställa in varan ifrån en leverantör då det utgått eller inte längre skall köpas in.
* Tillverkare – Ett indexeringsattribut. Det är bra att kunna filtrera och kategorisera produkter efter en viss Tillverkare och möjligheten att lagra denna data kan vara bra att ha för Bygghandelns utbud på deras hemsida.
* Kategori - Säger sig självt. Man behöver kunna filtrera alla produkter efter en viss kategori.
* SäljPris - Attributet står för ett slags rekommenderat säljpris för en produkt.
* Streckkod (EAN) – Byggvaruhus AB nämner i kundkonversationen att det hade varit användbart att ha en streckkod för varje produkt så de kan skanna när de utför inventeringar. Datalagring av streckkoder för varje produkt tillåter en att skriva ut streckkoder och uppfyller detta önskemål. Detta värde är unikt för varje rad i kolumnen för att säkerställa att varje produkt har ett unikt EAN-nummer.

För entiteten Leverantör har vi attributen:

* LeverantörID - Primärnyckel för varje instans av entiteten Leverantör
* Kontaktperson (flervärt attribut) - Det kan finnas fler än en kontaktperson för varje leverantör och det kan vara bra att veta vem man kontaktar när kontakt sker mellan Byggvaruhus och leverantören.
* Telefonnummer (flervärt attribut) - Ett telefonnummer behövs för att kunna kontakta leverantören.
* MailAdress (flervärt attribut) - Byggvaruhus AB nämner i kundkonversationen att de mailar inköpsordrar till sina leverantörer.
* Företagsnamn - Behövs för att kunna identifiera varje leverantör
* BankGiro – Bra information att ha då Inköpsfakturor dokumenteras i OLF-systemet.

För entiteten Kundorder har vi attributen:

* OrderNr - Primärnyckel för varje instans av en kundorder.
* OrderDatum - Behövs för indexering av kundordrar.
* TotalPris – varje orders totalpris behövs för att räkna ut Kundfakturans totalsumma.

För entiteten Stamkund har vi attributen:

* KundNr - Primärnyckel för varje instans av en stamkund.
* Företagsnamn - identifierar varje stamkundsinstans. (kundkonversationen nämner att endast företag kan bli stamkunder)
* FakturaAdress – Stamkunden behöver kunna få fakturor till sin företagsadress.

För entiteten Faktura har vi attributen:

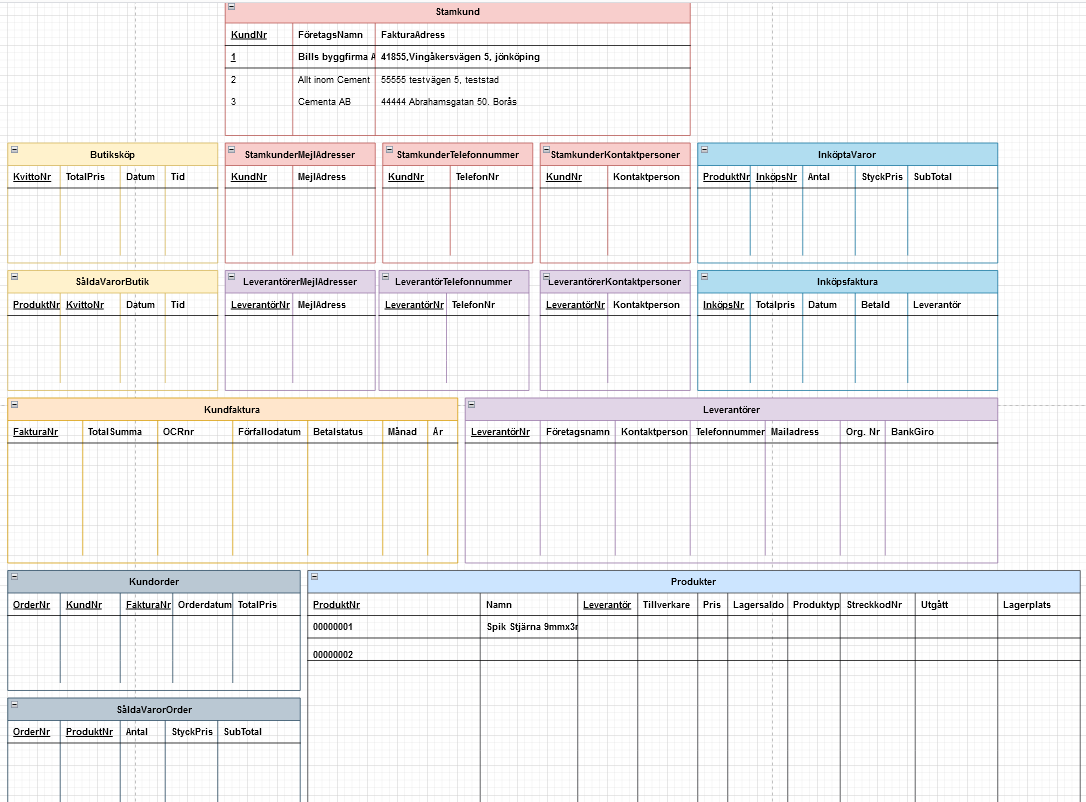
* FakturaNr - Primärnyckel för varje instans av en faktura.
* Totalsumma – Varje faktura har en totalsumma som behöver faktureras till kunden i slutet på månaden.
* OCRnr – Ett genererat OCRnummer för respektive faktura.
* Förfallodatum - Bra data att spara då det kan hända att kund tappar bort sin faktura och begär denna information.
* Betalstatus - Man kan programmera OLF-systemet att automatiskt mejla kunder som inte betalat sina fakturor efter att förfallodatumet passerat, eller helt enkelt bara filtrera efter obetalda fakturor.
* Månad - Månaden som fakturan gäller för. Kundfakturorna för Byggvaruhus AB samlar ihop kundordrar under en månad och fakturar kunden månaden efter.
* År - bra för indexering av fakturor.

För entiteten Butiksköp har vi attributen:

* KvittoNr - Primärnyckel för entiteten.
* Datum – datum för köpet.
* Tid – Den exakta tiden för köpet.

För entiteten Inköpsorder har vi attributen:

* InköpsNr - Primärnyckel för entiten.
* Totalpris – det totala priset kan vara bra att ha för varje inköpsorder. Det kan sedan användas för att räkna ut utgifter för respektive leverantör.
* Datum – datumet för inköpsordern kan kombineras med totalpris för att räkna ut utgifter per månad för respektive leverantör.
* Betalstatus – data om ordern är betald eller inte.



**Figur 2 - Tidig relationsmodell för Byggvaruhus AB.**

En relationsmodell är som ER-diagrammet en ytterligare typ av konceptuell datamodell, men med fördelen att vara en modell utav en relationsdatabas, vilket är den typen av databas som är dominerande i dagens databassystem. En relationsmodell går ut på att data lagras i relationer. En relation i detta sammanhang är samma sak som en tabell, med rader (som representerar varje instans av tabellen) och namngivna kolumner (som representerar de attribut som tillhör tabellen) (Databasteknik, 2005).

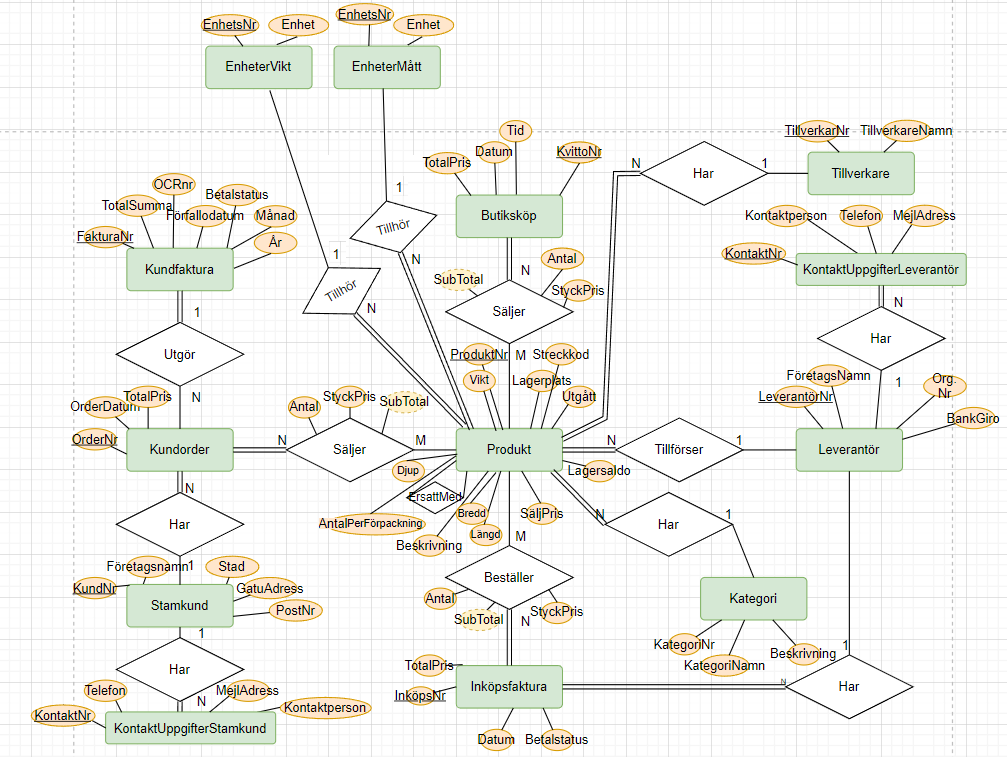
Teorin om normalisering kan appliceras på en relationsdatabas för att undvika dålig design (Padron-McCarthy & Risch, 2018). Oftast går det mesta i designen om man följer principen om en typ av sak per tabell, och en sådan sak per rad. Men ibland är det svårt att riktigt veta vad det är för "saker" man vill lagra, och vilka data som egentligen hör ihop med dem. Då har vi nytta av teorin om normalisering. Den hjälper oss att se exakt hur de olika kolumnerna i tabellen hör ihop, och visar oss hur vi ska dela upp tabellen för att slippa problemen (Padron-McCarthy & Risch, 2018). Därför börjar vi nu titta på de olika normalformer som teorin om normalisering beskriver.

Det enklaste villkoret är första normalformen, och genom att lägga på fler villkor kan man definiera andra normalformen, tredje normalformen, och så vidare (Padron-McCarthy & Risch, 2018). Första normalformen (1NF) uppnås genom att endast ett värde får finnas per ruta i tabellerna. Andra normalformen (2NF) uppnås genom att tabellerna inte får innehålla några fullständiga funktionella beroenden på delar av primärnyckeln, endast hela. Tredje normalformen (3NF) säger att inget icke-nyckelattribut får vara fullständigt funktionellt beroende av något annat icke-nyckelattribut (Padron-McCarthy & Risch, 2018).

Figur 2 visar en tidig relationsmodell som är baserad på första versionen av ER-diagrammet och är alltså inte den slutgiltiga konceptuella datamodellen. Här klargörs ett antal problem:

* Det går inte att hålla koll på vilket telefonnummer och mejladress som tillhör respektive kontaktperson då informationen för dessa tre attribut är uppdelade i tre separata tabeller.
* Kategori & Tillverkare behövs fyllas i om och om igen för varje instans av en Produkt.
* Attributet FakturaAdress i tabellen Stamkund uppnår inte 1NF.

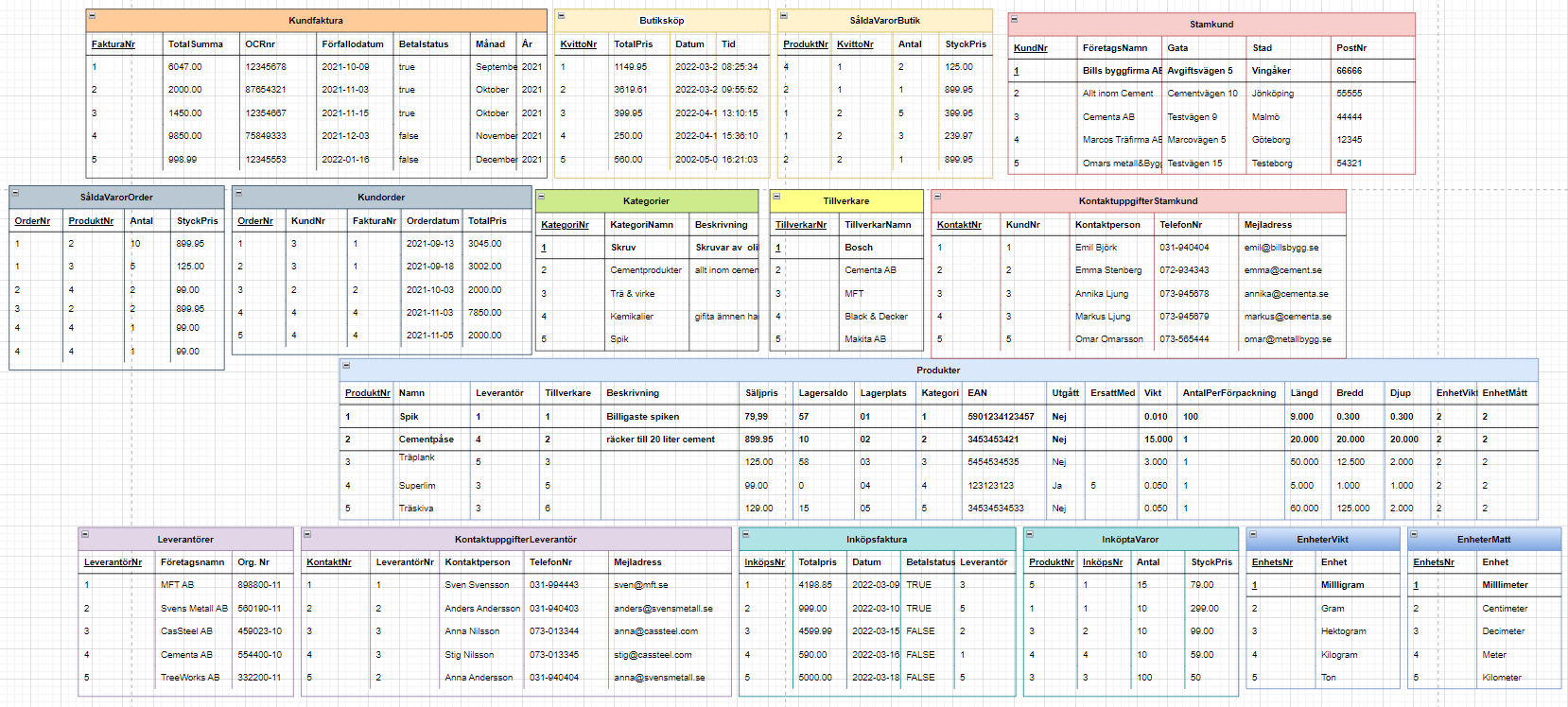
Därför reviderades ER-diagrammet till detta slutgiltiga resultat:



**Figur 3 – Slutgiltigt ER-diagram**

Ändringar utöver det första utkastet:

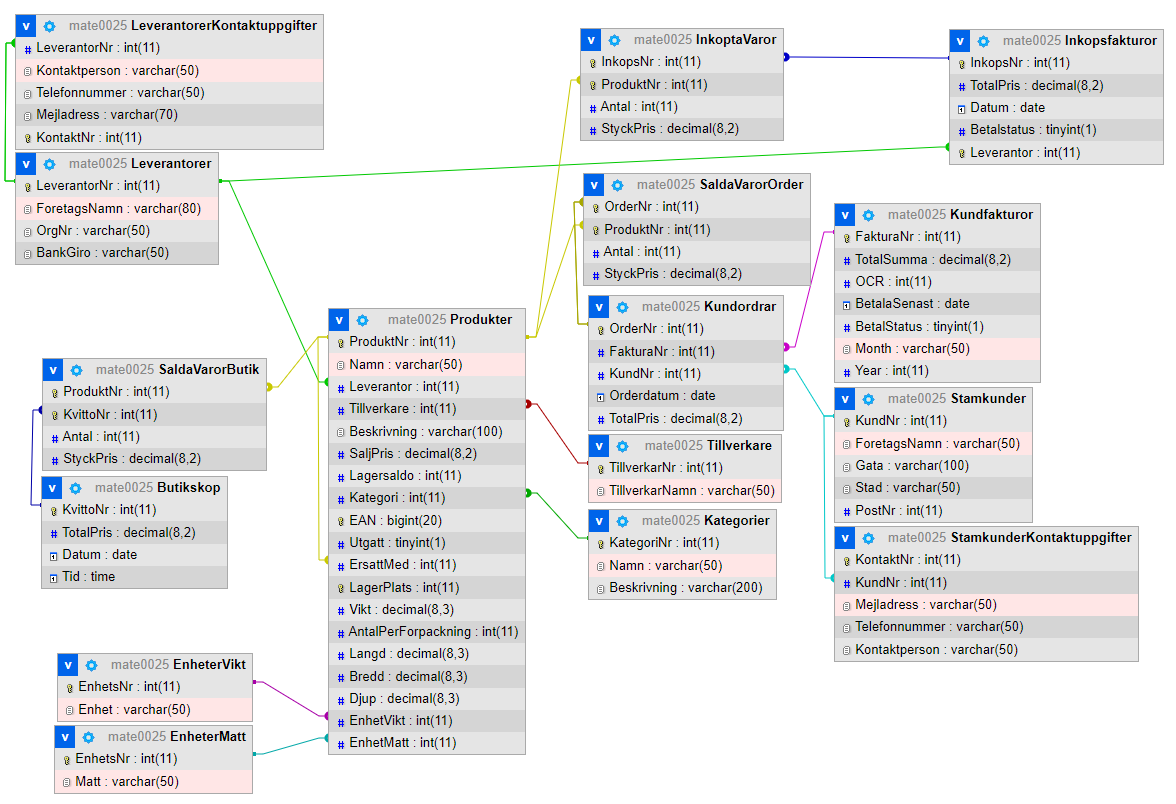
* Kategori finns nu som entitet istället för attribut i Produkt för att undvika redundans samt upprätthålla referensintegritet.
* Entiteterna KontaktUppgifterStamkund samt KontaktUppgifterLeverantör finns nu som entiteter med 1:N-samband till Stamkund istället för attribut i Stamkund respektive Leverantör. Anledningen till varför dessa attribut inte längre existerar som flervärda attribut är att det kan finnas två kontaktpersoner men endast ett telefonnummer till en specifik stamkund, vilket leder till att man inte vet vilken kontaktperson som telefonnumret tillhör.
* Tillverkare existerar nu som entitet istället för attribut för minskad redundans samt ökad referensintegritet. Faktum att Tillverkare är en separat entitet underlättar även indexeringen av produkter som tillhör en specifik tillverkare.
* Stamkund har nu attributen Stad, Gata och PostNr istället för FakturaAdress. Detta löser problemet med normalisering.
* Inköpsfaktura har nu ett starkt samband till Leverantör. En Inköpsfaktura kan ju inte existera utan att ha en Leverantör.
* TotalPris i sambandstabellerna heter nu SubTotal för att mer tydligt reflektera attributets data och existerar som härlett attribut i diverse entiteter istället för vanligt attribut. Detta attribut kändes meningslöst att spara i databasen gentemot verksamhetens storlek samt användarkriterier för OLF-systemets databas. Attributen Antal samt StyckPris tillåter en fortfarande att räkna ut en SubTotal för varje rad med inköpta varor med en Query om det så skulle behövas.
* Den tidigare entiteten Inköpsorder heter nu Inköpsfaktura för tydlighetens skull.
* En sambandsinstans ErsattMed finns nu för entiteten Produkt som komplement till attributet Utgått. Logiken bakom detta är att det kan vara bra att veta ifall en produkt har utgått och blivit ersatt med en annan. Då pekar attributet ErsattMed i en instans av en Produkt mot en annan instans av en Produkt och det signaleras med Romben ‘’ErsattMed’’ i diagrammet.
* Attributen Vikt, AntalPerFörpackning, Djup, Bredd och Längd finns nu med i entiteten Produkt då det är viktig information för varje produkt som gör det arbetet enklare i bygghandeln men också för att söka och filtrera efter dessa egenskaper.
* Två entiteter “EnheterMått” samt “EnheterVikt” har skapats med tillhörande attribut “EnhetsNr” samt “Enhet” för båda entiterna enligt figur 3. Dessa har skapats för att kunna ge fastställda måttenheter till produktattributen Vikt samt Djup, Bredd och Längd. Både EnheterMått samt EnheterVikt har ett 1:N-samband till Produkter-entiteten med ett fullständigt deltagande från Produktens sida då varje instans av en Produkt bör ha en enhet för både mått och vikt.



**Figur 4 – Slutgiltig Relationsmodell.**

Relationsmodellen i figur 4 speglar det slutgiltiga ER-diagrammet i figur 3 och uppfyller 1NF, 2NF och 3NF genom att inte ha mer än ett värde per ruta i tabellerna, alla fält i en tabell är direkt relaterade till nyckeln samt inte ha något referensattribut som pekar mot något annat än en kandidatnyckel, alltså ett attribut som är unikt för varje rad.

# Implementering



**Figur 5 – Designvy i PhpMyAdmin**

När ER-Diagrammet och Relationsmodellen är färdiga och uppnått 3NF-standarden så är det dags att realisera dem i PHPMyAdmin. Namnet på databasen är mate0025. Den innehåller 16 tabeller med 190 rader exempeldata totalt.

Det har varit riktigt kul att skapa en databas utifrån sin egen syn på verksamheten. Jag har behövt ändra mitt ER-diagram, relationsmodell och databas ungefär 10 till 15 gånger då jag insett att vissa attribut varit fel uttänkta eller inte matchat det som stått i mina konceptuella datamodeller, men det mesta har jag lyckats lösa. Det fanns en viss osäkerhet i om jag lyckats sätta rätt villkor och samband på alla kolumner i databasen, så som NOT NULL på kolumner som bör vara det och om jag har satt primärnycklar på kolumner som inte egentligen behöver det. Hade jag gjort om arbetet så hade jag kunnat tänka mig att göra om attributet LagerPlats i entiteten Produkt. Jag hade mycket hellre sett lagerplats som en separat entitet med ett 1:N-samband till Produktentiteten. På detta sätt hade jag kunnat ha flera kolumner för en hyllplats i formatet ‘’A-01-01'’ För att symbolisera hylla, hyllrad samt plats på hyllrad. Med tiden som fanns för arbetet räckte det inte till och varje hyllplats får så som datamodellen ser ut idag bara ett unikt värde så som 0,1,2 och så vidare. Lite otydligt tycker jag.

**SQL-sats 1:**

SELECT Kundordrar.OrderNr,Namn,Antal,StyckPris FROM SaldaVarorOrder,Stamkunder,Kundordrar,Produkter WHERE Stamkunder.ForetagsNamn='PEAB AB' AND Kundordrar.OrderNr=1 AND Stamkunder.KundNr=Kundordrar.KundNr AND Kundordrar.OrderNr=SaldaVarorOrder.OrderNr AND SaldaVarorOrder.ProduktNr=Produkter.ProduktNr;

**SQL-sats 2:**

SELECT Namn,Beskrivning,Lagersaldo FROM Produkter WHERE Lagersaldo<5 ORDER BY Lagersaldo DESC;

**SQL-sats 3:**

SELECT ForetagsNamn,Stad,Kundordrar.KundNr,Orderdatum FROM Kundordrar,Stamkunder WHERE Kundordrar.Orderdatum LIKE '2022-03%' AND Stamkunder.Stad LIKE 'Vänersborg' AND Kundordrar.KundNr=Stamkunder.KundNr;

# Transaktionshantering

Så som databasen är utformad så kommer verksamheten att behöva utföra transaktioner när butiksköp, kundordrar eller inköpsordrar hanteras då lagersaldot på inblandade produkter behöver minskas när produkter säljs.

**Transaktion 1:**

Ett butiksköp sker.

1. I tabellen Butikskop börjar transaktionen med att en instans med ett KvittoNr skapas utan värden för Tid och datum (dessa kan ha Null-värden).
2. SaldaVarorButik behöver ta detta kvittonummer och mata ihop med ett produktnummer för varje produkt som skall säljas samt antal och styckpris.
3. När köpet ‘’går igenom’’ i OLF-systemet och i samband med betalning så minskas varje produktnummers lagersaldo i tabellen Produkter med antal sålda produkter ifrån varje rad i tabellen SaldaVarorButik.
4. Samtidigt som köpet slutförs sätts också värden för Tid och Datum i tabellen Butikskop för kvittonumret.

**Transaktion 2:**

En kundfaktura genereras.

1. I tabellen Kundorder skapas en order.
2. I tabellen SaldaVarorOrder skapas en tabell för ordern där OrderNr och ProduktNr blir en primärnyckel för varje rad samt data om Antal och StyckPris.
3. När instanserna av SaldaVarorOrder är inmatade så behöver Ordern markeras som färdig i OLF-systemet.
4. För varje värde på attributet Antal som tillhör varje såld produkt för ordern så minskar lagersaldot för respektive produkt i tabellen Produkter.
5. Efter detta sparas ett totalpris och orderdatum i tabellen Kundorder.
6. En kundfaktura genereras med ett fakturanummer, totalsumma, OCR-nummer samt ett datum i kolumnen BetalaSenast som beräknas en månad framåt från datumet den första ordern gjordes för den månadens faktura. Även information om Månaden fakturan skapades lagras samt År.

Det är viktigt att skriva dessa operationer som en transaktion, så att egenskaperna Atomicity, Consistency, Isolation, Durability (ACID) upprätthålls (Padron-McCarthy & Risch, 2018). För de två transaktionerna betyder Atomicity (atomicitet) rent konkret att en transaktion måste kunna genomföras helt, eller inte alls. Upprätthålls inte atomicitetens egenskaper i en databas riskerar den att göra data korrupt (Padron-McCarthy & Risch, 2018). Vi kan ta Transaktion 2 som exempelscenario på vad som kan hända om inte atomicitetens egenskaper upprätthålls. Om databashanteraren kraschar när steg 3 är färdigt men steg 4 inte påbörjats så kommer värdet på attributet Lagersaldo inte ha hunnit subtraheras med värdet för Antal. Detta medför att Lagersaldot för en viss produkt kommer att vara inkonsistent. Det kommer att vara det samma som innan transaktionen startades trots att en instans av SaldaVarorOrder skapats med information om att ett visst antal av varan har sålts.

Egenskapen Consistency (konsistens) står för att databasen endast skall tillåta ändringar som leder till tillåtna tillstånd. Det vill säga att alla data som skrivs till databasen måste följa databasens regler (Padron-McCarthy & Risch, 2018). Upprätthålls inte konsistens inom databasen så kan det leda till interna motsägelser. Exempelvis kan en kolumn i en tabell vars datatyp (den typen av värde som skall sparas i kolumnen) är ett numeriskt värde “strunta” i att upprätthålla denna regel och ta in bokstäver istället. Egenskapen Isolation (isolering) säger att för att kunna fullgöra transaktioner gäller det att all härrörda data skall vara i ett tillåtet tillstånd (Databasteknik, 2005). Utan isolering kan exempelvis en transaktion i samma ögonblick komma åt en annan transaktions halvfärdiga ändringar. Med andra ord får en transaktions mellansteg inte påverkas av en annan transaktion under tiden den pågår.

Den sistnämnda egenskapen Durability (ihärdighet) står för att databasens data skall kunna bistå vid en eventuell krasch eller strömavbrott. Detta uppfylls med att data sparas på en hårddisk (Padron-McCarthy & Risch, 2018). Appliceras inte ihärdighetsegenskapen kan delar av data eller hela databasen förloras vid en eventuell krasch eller strömavbrott.

I min mening hade det i praktiken enkelt gått att undvika problem med ’deadlocks’ som kan uppstå när flera användare försöker läsa/skriva information till samma poster samtidigt genom att OLF-systemet hämtar mallen för SQL-syntaxen som härrör en viss transaktion och väntar att exekvera den fram tills att den butiksansvarige matar in relevant data och sedan kör hela transaktionen för att se så att inte någon operation slutförs innan allt garanterat är i ett tillåtet tillstånd. Skulle det mot förmodan uppstå en deadlock så kan man undvika det genom att sätta ett ‘timeout-villkor’ (Padron-McCarthy & Risch, 2018). Utöver att använda sig av SQL-kommandot COMMIT och ROLLBACK så använder sig databashanteraren av återhämtning och loggfiler för att återskapa databasen om den för någon anledning skulle krascha (Databasteknik, 2005).

# Distribuerade databassytem och Data Warehouse

För Byggvaruhus AB hade en klient-serverdatabas varit mest smidig för att undvika eventuella problem med nätverkshastigheter och komplexitet, men hade verksamheten haft fler än en lokal eller många noder så hade vi kunnat använda oss av en distribuerad databaslösning för att låta varje avdelning lagra sina data lokalt, men ändå kunna göra sökningar i hela verksamhetens datamängd (Databasteknik, 2005).

Ponera att det finns 3 noder i verksamheten Byggvaruhus AB:

* En nod i den ekonomiansvariges kontor
* En nod i butiken för den kassaansvarige
* En nod i lagret för lagerarbetarna

**På tabellnivå:**

* Tabellerna Inköpsfakturor & InkoptaVaror, Leverantörer & LeverantörerKontaktuppgifter hade kunnat lagras lokalt på den ekonomiansvariges nod då denna data kommer att hämtas och ändras utav denne.

*‘’Idag hjälps vi åt med de flesta av arbetsuppgifterna (bortsett från inköp och kommunikation med leverantörer, vilket endast jag sköter).’’ -* Ekonomiansvarig, Byggvaruhus AB.

* Tabellen Produkter hade kunnat fragmenteras vertikalt för att få deltabellerna Produkter\_1 & Produkter\_2 (se nedan).
* Tabellerna Stamkunder, StamkunderKontaktuppgifter, SaldaVarorOrder, SaldaVarorButik, Butikskop, Kundordrar, Tillverkare och Kategorier bör lagras på den butiksansvariges nod då denne kommer att behöva övergripande tillgång till dessa.

**På postnivå:**

* Om man tänker sig att lagerarbetarna har åtkomst till databasen från en nod i lagret så hade en vertikal fragmentering av tabellen Produkter vart användbar. Tabellen hade då fragmenterats till två:

**Produkter\_1:** ProduktNr, LagerPlats, EAN

Placering: Lagerarbetarnas nod.

**Produkter\_2:** ProduktNr,Namn,Leverantor,Tillverkare,Beskrivning,SaljPris,Lagersaldo,Kategori,Utgatt,ErsattMed

Placering: Den butiksansvariges nod.

Där Produkter\_1 Kan vara bra för lagerarbetarna om de behöver hämta eller ändra information om en produkts lagerplats och streckkodsnummer (EAN). Produkter\_2 kan användas av Butiksansvarige för att snabbt kunna få en överblick över lagersaldo och samspela med den ekonomiansvarige för vad som behöver beställas in.

Om Byggvaruhus AB hade varit en kedja med flera butiker så hade den konceptuella datamodellen kunnat anta en ytterligare entitet kallad Butik, där attributen hade kunnat vara ButiksID, Gata, PostNr, Stad och TelefonNr. Med hjälp av ett 1:N-samband till tabellerna Produkter och Stamkunder så hade ButiksID kunnat anta en kolumn i dessa tabeller. Det hade då vart fördelaktigt att utföra en horisontell fragmentering efter butiksnummer för att särskilja på stamkunder och produkter som tillhör en specifik butik.

Data mining är den process där stora mängder data samlas in och analyseras för att hitta mönster eller samband. Med den metadata (data om data) som skapas av datamining kan man exempelvis minimera risker, öka intäkter och minska utgifter och förbättra kundkontakten (SAS, 2022). Utifrån data mining-perspektiv så hade det gynnat verksamheten att få användbar statistik som till synes inte finns i databasen. Den data som hade kunnat sammanställas hade kunnat vara:

* Vilka produkter som säljs snabbast.
* En lista över vilka stamkunder genererar störst inkomst för Byggvaruhus AB.
* Vilka leverantörer som levererar billigast produkter inom en viss kategori.
* Vilken stad de flesta stamkunder kommer ifrån.
* Vilka produkter som har minst skillnad i Inköpspris kontra Säljpris.

Värdet av data mining är i min åsikt minst lika viktig som informationen i databasen.

# Big Data Ethics

Big Data refererar till data och metadata (data om data) där volymen, insamlingshastigheten samt variationen begränsar möjligheten att utföra effektiv analys med traditionella dataanalysmetoder (Richards & King, 2014).

Med prisutvecklingen av lagringskapacitet och utvecklingen av dataanalys så utvecklades mjukvaror såsom Hadoop för att massiva mängder data skulle kunna bearbetas snabbt för bland annat profilering av användare. Företag såsom Facebook, Google och Amazon ansluter sina tjänster till Hadoop för att samla in, bearbeta och sälja vidare denna information till tredjepartsföretag (Richards & King, 2014).

Trots att omfattande lagar finns kring personliga data så existerar det ännu inga sådana omfattande lagar för metadata. Personlig data kan dessutom ‘’anonymiseras’’ och därefter säljas på datamarknaden där den kan kombineras med annan relevant data för att ändå skapa en omfattande profilering (Richards & King, 2014). Därför granskar vi nu Byggvaruhus AB för att analysera vilka potentiella data som kan komma att ‘’läcka ut’’. Då verksamheten skall använda OLF-systemet som program installerat på en Windows-PC så kommer informationssystem så som Windows, Google och Facebook sannolikt att vara närvarande i verksamheten.

Vid användning av Googles tjänster så sparas även IP-adresser, sökfraser samt tidpunkt vilket kan ge metadata om vad Byggvaruhus AB har för verksamhet, hur verksamheten är uppbyggd samt vad de anställda har för intressen. Enligt Googles användarvillkor gäller: ‘We also collect the content you create, upload, or receive from others when using our services. This includes things like email you write and receive, photos and videos you save, docs and spreadsheets you create, and comments you make on YouTube videos.’ (Google, 2022). Google kan till exempel med hjälp av Google-sökningar ta reda på om en person nyligen varit med om ett dödsfall inom sin sociala krets, har psykologiska problem eller varit vittne till ett brott (Gallanan, Perri & Tomkowicz, 2021). Windows 10 skickar I sin fulla form information till Microsoft om bland annat webbläsarsökhistorik, platsinformation och installerade program (Microsoft, 2022). Enligt Facebooks användarvillkor gäller: “We collect the content, communications and other information you provide when you use our Products. It can also include what you see through features we provide, such as our camera.” (Facebook, 2022).

Eftersom lagarna kring metadata idag inte är lika omfattande som för personlig data så har företag I många fall rätten att bygga upp en profil med känslig information om en individ som kan köpas upp av återförsäljare, annonsörer, privatpersoner, ideella organisationer eller statliga myndigheter och beroende på kontext kan användas för allt ifrån marknadsföring till utpressning (Richards & King, 2014). I min mening så leder oetisk användning av Big Data till minskad acceptans och ökat samhällsmotstånd till även de positiva delarna av Big Data. Hamnar data I fel händer kan fundamentala demokratiska rättigheter som politiska åsikter tystas och trender kan stoppas. Datan skulle också kunna användas för oetiska saker som nätfiske och bedrägerier. Detta leder till att fråga sig; vad är rätt och fel? Big Data har blivit såpass stort att den med hjälp av korrekt insamlad och bearbetad data kan förutspå delar av framtiden, men betyder det att vi på grund av att vi har möjligheten till det bör försöka påverka den?

# Referenser

Databasteknik (2005-07-18). *ER-modellering*. [Elektronisk]. Örebro. Tillgänglig: <http://www.databasteknik.se/webbkursen/er/index.html> [2022-03-06].

Databasteknik (2005-07-18). *Relationsmodeller*. [Elektronisk]. Örebro. Tillgänglig: <http://www.databasteknik.se/webbkursen/relationer/index.html> [2022-03-10].

Databasteknik (2005-07-18). *Transaktioner*. [Elektronisk]. Örebro. Tillgänglig: <http://www.databasteknik.se/webbkursen/transaktioner/index.html> [2022-03-26]

Gallanan, G., Perri, D., Tomkowicz, S. (2021). Targeting vulnerable populations: The ethical implications of data mining, automated prediction, and focused marketing. [Elektronisk]. *International Journal of Organizational Innovation*. Spring2011, Vol. 3 Issue 4, p106-139. Tillgänglig: Business Source Ultimate [2022-03-26].

Google (2022-02-10). *Google Privacy Policy.* [Elektronisk]. Kalifornien. Tillgänglig: <https://policies.google.com/privacy?hl=en> [2022-03-26].

Framtidens Karriär (2022-02-18). *Polisen populäraste myndigheten - Framtidens karriär IT & Data.* [Elektronisk]. Tillgänglig: <https://framtidenskarriar.se/itdata/2022/02/18/polisen-popularaste-myndigheten/> [2022-03-26].

Microsoft (2022). Data collection summary for Windows. [Elektronisk]. Tillgänglig: [Data collection summary for Windows (microsoft.com)](https://privacy.microsoft.com/en-US/data-collection-Windows) [2022-04-24].

SAS (2022). *What is data mining? What it is and why it matters.* [Elektronisk]. Tillgänglig: <https://www.sas.com/en_us/insights/analytics/data-mining.html> [2022-03-27].

The Hacker News (2015-08-14). *Windows 10 Doesn't Stop Spying You, Even After Disabling It's Creepy Features.* [Elektronisk]. Tillgänglig: <https://thehackernews.com/2015/08/windows-10-privacy-spying.html> [2022-03-26].

Richards, N. & King, J. (2014). Big Data Ethics. [Elektronisk] *Wake Forest Law Review,* vol. 49, 393.Tillgänglig: SSRN [2022-04-24].

Padron-McCarthy, T. & Risch, T. (2018). *Databasteknik*. Stockholm: Studentlitteratur AB.