Del 1: ER-modellering og databaseteori

(a) ER-modellering

Oppgavens del 1a er delt i tre deler. I første del skal det lages et ER-diagram for databasen til firmaet Sjokoladeland. Deretter skal det lages et logisk skjema, og løsningen skal normaliseres.

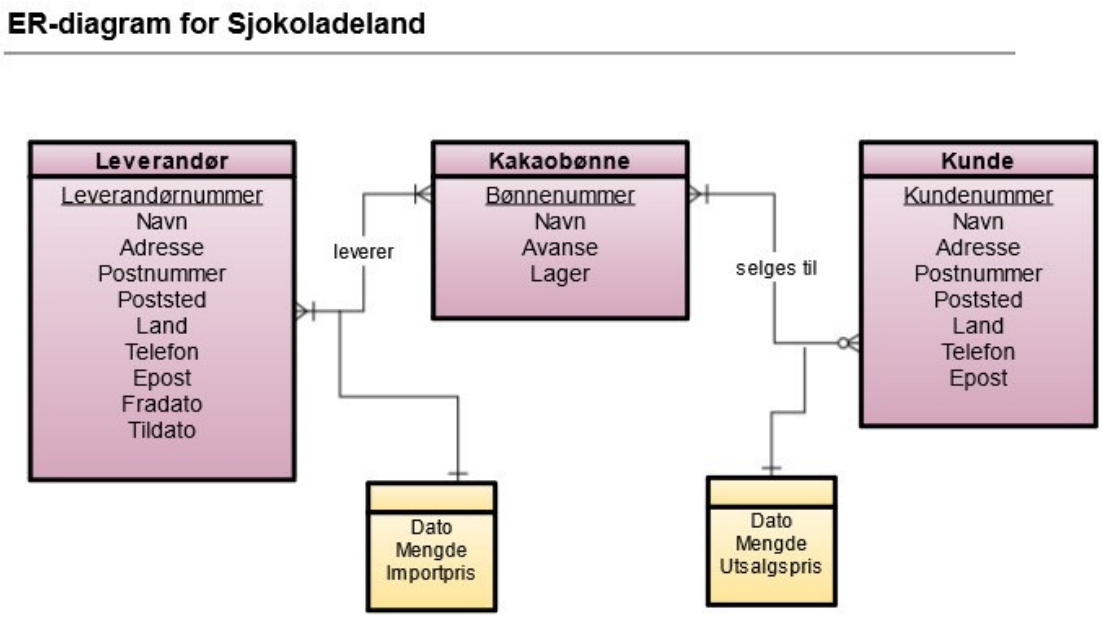
1. ER-diagram for Sjokoladeland

"ER" står for "Entity / Relationship" (Berget 2010 s. 13). Et ER-diagram er en grafisk fremstilling av strukturen i en database, men beskrivelsen er abstrakt, og tar ikke stilling til hvordan dataene skal representeres fysisk (Kristoffersen 2007 s. 128). ER-diagrammet utgjør dermed en slags visualisering av databasens funksjon og oppbygning, fremfor dens tekniske spesifikasjoner og kapasiteter. En ER-modell kan egne seg godt som diskusjonsgrunnlag i møter mellom sluttbrukere og databasedesignere, der man avklarer behov og begrepsbruk (Kristoffersen 2007 s. 128). Utviklingen av en database, også modelleringsfasen, må alltid ta utgangspunkt i hvordan og av hvem databasen forventes brukt (Berget 2010 s. 14), og man bør ha klart for seg hvilken informasjon man vil trenge å hente ut av databasen før man begynner designprosessen.

Sjokoladeland er en virksomhet som skal importere kakaobønner til Norge for videresalg til kunder i Norge. Virksomheten er under oppbygging, og databasen må kunne utvides og tilpasses for å imøtekomme nye behov som oppstår som følge av de forretningsstrategiske valgene som tas fremover.

Sjokoladeland ønsker å holde oversikt over sine leverandører og hvilke typer kakaobønner de leverer. Når Sjokoladeland bestiller kakaobønner fra en leverandør, ønsker de å kunne registrere bestillingen med opplysninger om hva og hvor mye som er bestilt, og når bestillingen er gjort. Dette samme gjelder når Sjokoladeland selger kakaobønnene videre til kundene sine. Generell informasjon om kundene må også kunne registreres.

Ovennevnte kan illustreres i følgende ER-diagram:



Diagrammet består av fem entiteter/entitetstyper. Entiteter er objekter som man ønsker å lagre informasjon om, enten fysiske "ting", som personer, ansatte og biler, eller mer abstrakte enheter, som ordrer, avtaler, prosjekter eller hendelser (Kristoffersen 2007 s. 129). For Sjokoladelands vedkommende, vil bønnen Criollo være en *entitetsforekomst* av entitetstypen "Kakaobønne". Bønnen Forasteros er en annen entitetsforekomst. Opplysninger om entitetsforekomstene må registreres av Sjokoladeland i databasen etter at den er ferdig utviklet og tatt i bruk.

Det er sammenhenger mellom de ulike entitetstypene i diagrammet. En leverandør kan levere mange typer bønner og en type bønne kan leveres av flere leverandører. Dette gir en "mange-til-mange"-sammenheng mellom entitetene "Leverandør" og "Kakaobønne". Sammenhengen er obligatorisk fordi en bønne alltid må leveres av noen (ellers er det ikke en bønnetype som tilbys av Sjokoladeland), mens en leverandør alltid må levere noe (ellers er han ikke en leverandør). En kunde kan kjøpe mange bønnetyper, og en bønnetype kan selges til flere kunder. Dette gir også en "mange-til-mange"-sammenheng, men fordi en bønne ikke må være solgt til en kunde, er sammenhengen til kunden gjort valgfri. (Sammenhengstyper diskuteres mer senere i besvarelsen.)

Fordi Sjokoladeland har behov for å holde styr på sine egne bestillinger fra leverandørene, i tillegg til bestillinger de mottar fra kundene, er sammenhengstypene, i likhet med de øvrige entitetene, tilknyttet *attributter*. Et attributt er en egenskap ved en entitetstype (Berget 2010 s. 15). Behovet for attributter på hver entitet bør analyseres grundig før databasen bygges. Samtidig bør man unngå å registrere attributter på entitetene som ikke er relevante for virksomheten som skal bruke databasen, fordi en slik praksis vil gjøre databasen unødig stor og full av data som aldri blir brukt til noe (Berget 2010 s. 15).

Attributtene i Sjokoladelands database er basert på Sjokoladelands antatte informasjonsuthentingsbehov etter at databasen er tatt i bruk. Fordi Sjokoladeland har antydet at det på sikt kan være aktuelt å utvide virksomheten, har entitetstypen "Kakaobønne" fått attributtet "Lager", slik at det vil være enkelt for Sjokoladeland å registrere hvilket lager bønnetypen oppbevares i, skulle virksomheten etter hvert få flere lager. Attributtet "Lager" kunne også vært tilknyttet sammenhengene, men det er antatt at hvor bønnene håndteres av Sjokoladeland snarere vil variere ut fra bønnetype (enkelte bønnetyper trenger kanskje spesialisert behandling, spesielle temperaturforhold eller annet), fremfor fra bestilling til bestilling. Dette er selvfølgelig ikke gitt, fordi Sjokoladeland kan ønske å utvide med lager som er lokalisert nærmere bestemte leverandører eller kunder for å redusere sine fraktkostnader, eller forsterke sin miljøprofil. Tilsvarende gjelder for attributtet "Avanse". Det antas her at Sjokoladeland ønsker seg en automatisert beregning av utsalgspris basert på en predefinert avansekoeffisient for hver enkelt bønne, altså at avansen varierer med bønnetypen alene, men Sjokoladeland kan også ha en annen prispolitikk, f.eks. at enkelte kunder får spesialpriser, eller prisene beregnes ut fra mengden bønner kunden kjøper. Hvordan man ønsker å registrere lager- og avanseinformasjon er dermed gode eksempler på detaljer som det vil være nødvendig å diskutere med sluttbrukeren av databasen tidlig i designprosessen, før databasen utvikles og tas i bruk.

Entiteten "Leverandør" er gitt datoattributter, slik at Sjokoladeland skal kunne registrere at samarbeidet med en leverandør er avsluttet, uten å slette all informasjon om leverandøren. Det samme vil man naturligvis kunne gjøre for kundeentiteten, slik at man f.eks. kan avslutte et kundeforhold pga. manglende betalingsevne, men det antas her at slik oppfølging gjøres av firmaet som håndterer Sjokoladelands fakturaoppgaver.

Bestillinger, både fra leverandører og fra kunder, må lagres med informasjon om dato og mengde. Disse attributtene får ikke verdier før det finnes en faktisk kobling – en transaksjon – mellom to entiteter (Berget 2010 s. 20). Attributtene kan derfor ikke med hell plasseres på (de statiske) entitetene. Bestillingene er gitt attributtene importpris og utsalgspris, da det, som nevnt, antas at dette kan variere, og dermed ikke kan knyttes direkte verken til den enkelte leverandør eller til den enkelte bønnetypen.

2. Logisk skjema for Sjokoladeland

Logisk databasedesign innebærer å oversette ER-diagrammer til logisk tabellstruktur, enkelt sagt at entiteter blir til tabeller (relasjoner), attributter til kolonner, identifikatorer til primærnøkler og forhold til fremmednøkler (Kristoffersen 2007 s. 153), eller nye relasjoner. Det logiske skjemaet er like teknologiuavhengig som ER-diagrammet, men det er implementerbart, dvs. at man kan opprette, definere og strukturere databasen på grunnlag av skjemaet (Berget 2010 s. 33).

Sjokoladelands logiske skjema ser slik ut (primærnøkler er understreket og fremmednøkler kursivert):

Leverandør (Leverandørnummer, Navn, Adresse, *Postnummer*, *Poststed*, Telefon, Epost, Fradato, Tildato)
Kakaobønne (Bønnenummer, Navn, Avanse, Lager)
Kunde (Kundenummer, Navn, Adresse, *Postnummer*, *Poststed*, Telefon, Epost)
Poststed (Postnummer, Poststed)
Land (*Poststed*, Land)
Leveranse (*Leverandørnummer*, *Bønnenummer*, Dato, Mengde, Importpris)
Salg (*Kundenummer*, *Bønnenummer*, Dato, Mengde, Utsalgspris)

Ulike sammenhengstyper konverteres ulikt. "Mange-til-mange"-sammenhenger blir til nye relasjoner i skjemaet. Fordi Sjokoladeland har to "mange-til-mange"-sammenhenger i sitt ER-diagram, får vi to nye relasjoner, "Leveranse" og "Salg", med tilhørende attributter.

En primærnøkkel er ett attributt eller flere attributter i kombinasjon, som gir en unik og entydig identifikasjon av hver entitetsforekomst (Berget 2010 s. 16). I Sjokoladelands logiske skjema er primærnøklene fra ER-diagrammet i de tre første relasjonene med. I de

to nye relasjonene må det velges primærnøkler. For at disse skal være en unike og entydige identifikatorer, må de i begge tilfellene settes sammen av flere attributter. Fordi man kan anta at det er unaturlig å bestille samme bønnetype flere ganger samme dag fra samme leverandør, og at dette også er unaturlig i andre enden, altså at en kunde bestiller den samme bønnetypen flere ganger samme dag, kan primærnøklene settes sammen henholdsvis av attributtene leverandørnummer, bønnenummer og dato for leveranse, og kundenummer, bønnenummer og dato for salg.

3. Normalisering

Normaliseringsteori angir kriterier for å avgjøre om en gitt tabellstruktur er god eller dårlig, og en stegvis prosess for å dele opp tabeller i flere enklere tabeller på en måte som fjerner redundans (dobbeltlagring), og samtidig bevarer informasjonen (Kristoffersen 2007 s. 153). Redundans gjør at databasen krever mer lagringsplass enn nødvendig, og en ikke-normalisert tabellstruktur øker sjansene for at informasjonen i databasen blir inkonsistent (Kristoffersen 2007 s. 152). Det finnes seks normalformer, men de tre første er de viktigste (Berget 2010 s. 49). Første normalform (1NF) oppnås når en tabell kun inneholder atomære verdier, andre normalform (2NF) oppnås når den i tillegg ikke inneholder partielle avhengigheter, dvs. at alle kolonner som ikke er en del av primærnøkkelen er funksjonelt avhengige av hele primærnøkkelen og ikke bare deler av den, og tredje normalform (3NF) oppnås når alle kolonner som ikke er en del av primærnøkkelen er gjensidig uavhengige (Kristoffersen 2007 s. 167-168 og Berget 2010 s. 49-54).

Jeg vil videre kun undersøke om Sjokoladelands datamodell oppfyller kravene tom. tredje normalform. I Sjokoladelands tabeller skal man utelukkende kunne legge inn atomære verdier, og løsningen er dermed i første normalform. Tabeller som har primærnøkler som kun består av én kolonne, er automatisk i andre normalform (Berget 2010 s. 51). Dette gjelder for alle relasjonene med unntak av "Leveranse" og "Salg", og det må derfor undersøkes hvorvidt alle kolonner som *ikke* er en del av primærnøkkelen i disse relasjonene, er funksjonelt avhengig av *hele* primærnøkkelen, og ikke bare deler av den. Ser vi på attributtet "Mengde" i begge relasjonene, er denne direkte knyttet til den aktuelle forekomsten av leveransen eller salget. Det samme gjelder prisen, da jeg har antatt at denne verdien vil kunne variere fra leveranse til leveranse/salg til salg. Løsningen er dermed i andre normalform. Da gjenstår det å vurdere hvorvidt løsningen er

i tredje normalform, altså om det finnes transitive avhengigheter mellom noen av kolonnene som ikke er en del av primærnøkkelen i hver relasjon. Man kan anta at verdien i "Mengde"-feltet i relasjonene "Leveranse" og "Salg" kan (bidra til) å bestemme prisen, og muligens også omvendt (altså at hvis man handler for over en viss sum, får man (automatisk) noen ekstra bønner med på kjøpet), men fordi dette vil være betinget av Sjokoladeland og Sjokoladelands leverandørers prispolitikk, antar jeg at dette er en i hovedsak illusorisk avhengighet, altså at det ikke har noen reell betydning her. Disse to relasjonene er dermed i tredje normalform. Tilsvarende gjelder for relasjonen "Kakaobønne". Ingen av verdiene i "Navn", "Avanse" eller "Lager" vil på noe tidspunkt entydig bestemme verdien i de øvrige feltene. I relasjonene "Leverandør" og "Kunde" blir dette mer komplisert fordi disse relasjonene inneholder adresseinformasjon. Verdien i "Postnummer" vil alltid bestemme verdien for "Poststed", altså er "Postnummer" determinant for "Poststed", som igjen er funksjonelt avhengig av "Postnummer" (Berget 2010 s. 55). Dette medfører at kravet til tredje normalform ikke er oppfylt. Tilsvarende gjelder for "Land", som vil være funksjonelt avhengig av "Poststed". For å oppnå en løsning som er i tredje normalform, er disse verdiene derfor skilt ut i egne relasjoner, som hver oppfyller kravene til tredje normalform.

Hele Sjokoladelands datamodell er dermed i tredje normalform, og kan implementeres i et databasehåndteringssystem.

(b) Databaser

Oppgavens del 1b består av tre deler. I første del skal det drøftes hvorfor bruken av databaser er viktig fra arkivarens perspektiv. I andre del skal det redegjøres for funksjonene til det som i databaseteorien kalles primær- og fremmednøkler. Deretter skal forskjellene mellom og funksjonene til de ulike sammenhengstypene "en-til-en", "en-til-mange" og "mange-til-mange", diskuteres.

1. Databaser fra arkivarens perspektiv

I teorien kunne nok de fleste virksomheter klart seg med manuelle systemer (Kristoffersen 2007 s. 4). Det samme gjelder muligens også for (virksomhets)arkiver. I praksis er det annerledes. Store datamengder, krav til sikkerhet, effektivitet og

tilgjengelighet gjør at manuelle systemer ikke lenger innfrir kravene som leverandører, kunder, ansatte og publikum har.

Databaser og digital tilgang gir helt nye muligheter for frem- og gjenfinning av informasjonen. Et godt eksempel på dette er oslobilder.no, som er et resultat av et samarbeid mellom flere kulturinstitusjoner og virksomheter i Norge. Databaser (i arkivsammenheng) dreier seg dermed vel så mye om å bedre mulighetene for tilgang til dataene, som til struktureringen av dem, selv om sistnevnte har lett for å komme i fokus.

EU anslo i 2011 at viderebruk av data kan utløse gevinster på opp mot 40 milliarder Euro (Mandag Morgen 2013 s. 76). Dette er ikke mulig hvis dataene ikke er lagret og strukturert slik at de kan hentes ut, tolkes effektivt og settes sammen på nye måter. Arkivarer bør bidra til denne utviklingen – og veksten – og samtidig være pådrivere for at dataene som brukes også gjøres tilgjengelige for ettertiden. Det ligger imidlertid litt i (mange) arkivarers "natur" å forsøke å bremse litt når man ikke kan være sikker på (de langsiktige) konsekvensene av ulike valg. Der IT-folk og dataanalytikere gjerne utelukkende ser nå-perspektivet, tenker arkivarene, med sin spesialiserte kompetanse og tilnærming til problemene, gjerne lenger frem. Det gir en unik mulighet til å bidra til å styre utviklingen slik at også kommende generasjoner kan ha nytte av informasjonssamlingene våre – også de digitale.

Den teknologiske utviklingen skjer uavhengig av enkeltprofesjoner. Hvis arkivarene skal kunne ordne, bevare og tilgjengeliggjøre arkivinformasjon, må også informasjonen som lagres i databaser (med eller uten arkivarens innvirkning) inkluderes. Utfordringen ligger i å finne de gode løsningene (som ivaretar flere behov på en gang), og å markedsføre arkivperspektivet på en slik måte at dette ikke feies til side for enkelhets skyld.

2. Databaseteori: primær- og fremmednøkler

Integritetsreglene i relasjonsmodellen utledes av teorien om bruken av primærnøkler og fremmednøkler (Kristoffersen 2007 s. 117). Enhver relasjon skal ha nøyaktig én primærnøkkel, og primærnøkkelen skal ikke inneholde nullmerker (tomme felter) eller to like verdier, dvs. at primærnøkkelen skal utgjøre en unik identifikator for en entitetsforekomst (Kristoffersen 2007 s. 118, 130). Disse to kravene bestemmer relasjonens entitetsintegritet, og sjekkes av databasehåndteringssystemet når en ny rad

settes inn i en tabell eller når en primærnøkkelverdi blir oppdatert. Som nevnt tidligere, vil Sjokoladelands database ikke godta at en kunde forsøker å kjøpe samme type bønne flere ganger på samme dato, fordi det ved bestilling nr. 2 allerede vil være registrert en entitetsforekomst av "Salg" med den aktuelle primærnøkkelen. Dette kan synes som en unødvendig begrensning av funksjonaliteten og mulighetene i databasen, og at primærnøkler vil øke forekomsten av feilmeldinger, noe som lett kan oppleves som negativt (Kristoffersen 2007 s. 56). Samtidig vil dette hjelpe oss å unngå feil, og dermed bidra til å bedre kvaliteten på dataene (Kristoffersen 2007 s. 56).

Fremmednøkler ivaretar koblingene mellom de ulike relasjonene i databasen. En fremmednøkkel er et attributt i én relasjon som refererer til en annen relasjon der attributtet er primærnøkkel (Berget 2010 s. 46). Fremmednøkkelen vil dermed alltid peke på en primærnøkkel som som regel er i en annen tabell (Kristoffersen 2007 s. 63). Fremmednøkler kan være sammensatte, kan inneholde nullmerker, og blir kontrollert av databasehåndteringssystemet når nye verdier settes inn i fremmednøkkelen og ved sletting av verdier i tilhørende primærnøkkel (Kristoffersen 2007 s. 119). Fremmednøklene settes inn i relasjonene når de ulike sammenhengstypene i ER-diagrammet konverteres til det logiske skjemaet. Dette er tilfellet i Sjokoladelands "Leveranse"- og "Salg"-relasjoner. I tillegg har Sjokoladeland fremmednøkler i relasjonene "Postnummer" og "Poststed" fordi disse måtte skilles ut fra relasjonene "Leverandør" og "Kunde" for at løsningen skulle være i tredje normalform. Fremmednøkler er dermed nært knyttet til de ulike sammenhengstypene og hvordan disse representeres i logisk databasedesign.

3. Databaseteori: sammenhengstyper

I relasjonsdatabaseteorien er en sammenheng en kobling mellom to entitetstyper, og uttrykker at entiteter av én entitetstype kan være relatert til en eller flere entiteter av en annen entitetstype og motsatt (Berget 2010 s. 17). Kardinaliteten til sammenhengen uttrykker hvor mange entitetsforekomster av entitet A som kan eller må knyttes til en gitt forekomst av entitet B, og omvendt (Kristoffersen 2007 s. 133). Det er tre hovedkategorier sammenhengstyper; "en-til-en"-, "en-til-mange"- og "mange-til-mange"-sammenhenger.

"En-til-en"-sammenhengstypen forteller at en entitetsforekomst A kun kan eller må kobles til én entitetsforekomst B, og motsatt. Dette hadde vært tilfellet for Sjokoladeland

dersom en leverandør kun kunne levere én type kakaobønne, og denne kakaobønnen kun kunne leveres av den aktuelle leverandøren. "En-til-mange"-sammenhengstypen forteller at en entitetsforekomst A kan være knyttet til mange B-forekomster, men hver B-forekomst vil være knyttet til maksimalt én A-forekomst (Kristoffersen 2007 s. 133). En kunde kan f.eks. ha mange bestillinger, men en bestemt bestilling kan kun gjøres av én kunde. "Mange-til-mange"-sammenhenger oppstår når entitetsforekomst A kan være knyttet til mange entitetsforekomster B, og omvendt. Slike sammenhenger har Sjokoladeland både mellom leverandør og kakaobønne og mellom kunde og kakaobønne.

Når sammenhengene mellom entitetene skal representeres på logisk nivå, blir "mange-til-mange"-sammenhengene til nye relasjoner, gjerne kalt assosiative entiteter eller koblingsentiteter, "en-til-mange"-sammenhenger blir til fremmednøkler på mange-siden og "en-til-en"-sammenhenger blir til fremmednøkler i en av relasjonene. De ulike sammenhengene kan videre være obligatoriske (vist med en "|" på den obligatoriske siden av sammenhengen), eller ikke-obligatoriske (vist med en "o" på den ikke-obligatoriske siden av sammenhengen). Sammenhenger mellom entiteter bør i tillegg gis forklarende navn, og noen ganger må de tilknyttes attributter, slik som i Sjokoladelands datamodell.

Kilder

Berget, G. (2010). *Relasjonsdatabaser og datamodellering 3. utgave.* Oslo: Høgskolen i Oslo og Akershus.

Kristoffersen, B. (2007). *Databasesystemer*. Oslo: Universitetsforlaget.

Mandag Morgen. (2013). *Big Data. Fremtidens digitale råstof.* Tilgjengelig 9. september 2013 via https://www.mm.dk/intelligence-big-data