cphbusiness

Modellering af data

Modellering af verden – én tilgang

Navn	Adresse	Vare 1	Vare 2	Vare 3	Vare 4	Pris 1	Pris 2	Pris 3	Pris 4	Antal 1	Antal 2	Antal 3	Antal 4
Signe	Byvej 4	Kiks	Æg	Mælk		13,5	25,95	10,00		4	1	2	
Kurt	Nygade 5	Æble				3,00				10			
Dorte	Højmarken 10	Banan	Æble			2,5	3,00			8	2		
Niels	Skolevænget 1	Brød	Smør	Kage	Juice	27,50	11,95	43,00	18,75	2	1	2	4

Her er en model af nogle kunder og deres ordrer.

Men der er nogle udfordringer ved modellen

- Der er mange huller, så tabellen fylder mere end den behøver
- Nogle værdier skal være de samme (fx prisen på æbler i række 1 og række 3), men der er ikke noget, der forhindres os i at lave prisen forskellig for samme vare
- Ingen kunder kan købe mere end fire varer af gangen, da der kun er lavet kolonner til fire varer
- Vi kan blive nødt til at bygge tabellen ud med kolonner til kundenummer, telefonnummer, mail, osv. Den kan blive kæmpestor og dermed ret uoverskuelig

Der er dog én fordel; når vi først har fundet en kunde i tabellen, så ved vi ALT om dem!

Modellering af verden – en bedre tilgang

Tabel: Customer

Id	Navn	Adresse
11	Signe	Byvej 4
12	Kurt	Nygade 5
13	Dorte	Højmarken 10
14	Niels	Skolevænget 1

Tabel: Order*

Id	Ordrenr	Kunde
21	AB3711	11
22	NY3488	12
23	MC5672	13
24	CK78654	14

Mange tabeller med simple datarelationer frem for én tabel med mange data.

- Opfylder disse tabeller 1. normalform? Hvorfor (ikke?)
- Opfylder de 2. normalform? Hvorfor (ikke)?
- Og opfylder de 3. normalform? Hvorfor (ikke)?

Tabel: Orderline

Id	Vare	Antal	Ordre
41	31	4	21
42	35	1	21
43	36	2	21
44	37	2	24

Tabel: Item

Id	Vare	Pris
31	Kiks	13,5
32	Æble	3,00
33	Banan	2,50
34	Brød	27,50
35	Æg	29,95
36	Mælk	10,00
37	Kage	43,00
38	Smør	11,95
39	Juice	18,75

* Note: Order er et dårligt tabelnavn, da det er et keyword i SQL

Relationer og kardinalitet

Tabel: Customer

Id	Navn	Adresse
11	Signe	Byvej 4
12	Kurt	Nygade 5
13	Dorte	Højmarken 10
14	Niels	Skolevænget 1

- Vi kan ikke længere vide alt, ved at slå op i én tabel.
- Vi benytter os af fremmednøgler.
- Relationer tager tid og komplicerer opslag (joins)
- Hvad er relationerne mellem tabellerne?
- Kan vi have alle relationer (1:1, 1:0, *:*, 1:* osv)?

Tabel: Item

Id	Vare	Pris
31	Kiks	13,5
32	Æble	3,00
33	Banan	2,50
34	Brød	27,50
35	Æg	29,95
36	Mælk	10,00
37	Kage	43,00
38	Smør	11,95
39	Juice	18,75

Tabel: Order

Id	Ordrenr	Kunde
21	AB3711	11
22	NY3488	12
23	MC5672	13
24	CK78654	14

Tabel: Orderline

Id	Vare	Antal	Ordre
41	31	4	21
42	35	1	21
43	36	2	21
44	37	2	24

1-1, 1-M og M-M

- 1-1 er let at lave
- 1-M: hvad er 1 og hvad er M?
- M-M: kræver en helt ny tabel (koblingsentitet). Hvorfor?
- Hvordan adskiller dette sig fra relationer i Java?

Entitet

- En entitet er et objekt/en række i databasen
- Kan sammenlignes med objekter i Java
- Repræsenterer noget, der findes i virkeligheden
- Kan identificeres vha. en unik identificer/primærnøgle

Structured Query Language (SQL)

- SQL kan bruges til at oprette tabeller, finde data, læse data, indsætte rækker, ændre data i en række, slette rækker, opdatere en eller flere rækker og meget mere. Basisoperationerne kaldes CRUD (*create, read, updata, delete).*
- En række i en database har felter, som fx id, navn og adresse for en kunde
- Vi beder vha. SQL om de data, der står i bestemte felter, i bestemte rækker, i en bestemt tabel fx

```
SELECT Name FROM Customer WHERE Id =’11’
```

- Det er en konvention, at SQL-ordene skrives med store bogstaver og navne på felter og tabeller skrives med små bogstaver. Det gør forespørgslen lettere at læse for trænede øjne.
- Vi kan også vælge at få alle data om vores kunde ved at bruge * , som betyder ”det hele”:

```
SELECT * FROM Customer WHERE Id =’11’
```

Opsamling

- Hvad er 1. normalform og hvorfor har vi brug for den?
- Hvad er 2. normalform og hvorfor har vi brug for den?
- Hvad er 3. normalform og hvorfor har vi brug for den?
- Hvad bruger vi ER diagrammer til?
- Hvem er målgruppen for ER diagrammer?
- Hvornår i vores udviklingsproces bruger vi ER diagrammer?

Domænemodel vs EER

- Domænemodel = den virkelige verden
- EER = databaseverden
- (Klassediagram = Javaverden)