Database design for begyndere

Denne artikel beskriver hvordan man kommer fra ide til database design.

Den stopper inden normal former.

Den forudsætter kun lidt kendskab til database terminologi og SQL.

Skrevet den **16. Feb 2010** af **arne_v** I kategorien **Databaser / Generelt** | ★★★☆☆

Historie:
V1.0 - 13/01/2004 - original
V1.1 - 31/01/2004 - forbedret formatering
V1.2 - 09/02/2004 - fixe et par små formaterings fejl
V1.3 - 25/07/2004 - tilføj lidt forklaring på index
V1.4 - 16/02/2010 - smårettelser

Procedure

Når man skal igang med sine første databaser er det ofte uhyggeligt svært at vælge en tabel struktur.

Når man har prøvet den en 20-40 gange, så gør man det bare.

Men man skal jo starte på et tidspunkt.

Jeg vil forsøge at tage lidt af magien ud af den process at finde en acceptabel database struktur.

Processen er:

1) find de objekt typer der skal gemmes i databasen og lav en tabel for hver af dem

2) find attributterne på hvert objekt og lav et felt for hver af dem

3) tilføj et id felt til tabeller hvor der ikke er en unik identifikation af forekomster blandt de eksisterende felter

4) find relationerne mellem objekterne

5) tilføj et ekstra felt på M siden af 1:M relationerne

6) tilføj en ekstra tabel med 2 felter for M:M relationerne

7) marker primær nøgler som unikt identificerer forekomster og fremmed nøgler som genviser til en anden tabel

8) vælg data type for alle felter

9) find de mest naturlige queries og sæt index på de felter der bruges der

Eksempel

Det var måske lidt rigeligt med fremmedord.

Men lad os tage et lille simpelt eksempel til at illustrere metodikken.

Lad os sige at vi skal lave en lille skole database.

1) find de objekt typer der skal gemmes i databasen og lav en tabel for hver af dem

Der er 3 meget oplagte objekter:
Elev
Klasse
Lærer

Så vi laver en tabel for hver af dem.

Tabel struktur:

```
Elev
----

Klasse
------

Lærer
-----
```

2) find attributterne på hvert objekt og lav et felt for hver af dem

Elev har følgende oplagte attributter:
Navn
Alder

Klasse har følgende oplagte attributter:
Navn

Lærer har følgende oplagte attributter:
Navn

Vi laver felter for hver af dem.

Tabel struktur:

```
Elev
----
```

```
Navn
Alder

Klasse
------
Navn

Lærer
-----
Navn
```

3) tilføj et id felt til tabeller hvor der ikke er en unik identifikation af forekomster blandt de eksisterende felter

Klasse navne er unikke, men både elever og lærer kan have mere end en med samme navn, så vi tilføjer et id felt til de tabeller.

Tabel struktur:

```
Elev
----
ID
Navn
Alder

Klasse
------
Navn

Lærer
-----
ID
Navn
```

4) find relationerne mellem objekterne

Vi ser at Klasse-Elev er en 1:M relation, da en klasse indeholder mange elever men en elev kun er i en klasse.

Vi ser at Klasse-Lærer er en M:M relation, da en lærer har mange klasser og en klasse har mange lærer.

5) tilføj et ekstra felt på M siden af 1:M relationerne

Tabel struktur:

```
Elev
----
ID
Navn
Alder
Klassenavn
```

```
Klasse
------
Navn

Lærer
-----
ID
Navn
```

6) tilføj en ekstra tabel med 2 felter for M:M relationerne

Tabel struktur:

```
Elev
----
ID
Navn
Alder
Klassenavn

Klasse
------
Navn

Lærer
-----
ID
Navn

LærerKlasse
-----------
LærerID
Klassenavn
```

7) marker primær nøgler som unikt identificerer forekomster
og fremmed nøgler som genviser til en anden tabel

Jeg vil bruge de engelske forkortelser.

PK = primary key
FK = foreign key

Tabel struktur:

```
Elev
----
ID (PK)
Navn
Alder
Klassenavn (FK)

Klasse
------
```

```
Navn (PK)

Lærer
-----
ID (PK)
Navn

LærerKlasse
-----------
LærerID (delt PK,  FK)
Klassenavn (delt PK, FK)
```

8) vælg data type for alle felter

Som hovedregel vælger man:
- INTEGER til tal
- VARCHAR til tekst
- NUMERIC Til beløb

og undgår:
- FLOAT
- CHAR

Bemærk at både udvalget af data typer og deres præcise egenskaber er database specifikke, men ovenstående er standard.

Tabel struktur:

```
Elev
----
ID (PK) - INTEGER
Navn - VARCHAR(30)
Alder - INTEGER
Klassenavn (FK) - VARCHAR(10)

Klasse
------
Navn (PK) - VARCHAR(10)

Lærer
-----
ID (PK) - INTEGER
Navn - VARCHAR(30)

LærerKlasse
-----------
LærerID (delt PK,  FK) - INTEGER
Klassenavn (delt PK, FK) - VARCHAR(10)
```

9) find de mest naturlige queries og sæt index på de felter der bruges der

Index er noget som gør det meget hurtigere at finde en bestemt værdi i et felt. Og de betyder meget for hastigheden af queries og derfor er det vigtigt at få index på de felter der skal have det.

Man bør derfor finde de mest naturlige queries (de queries som man forestiller sig vil blive udført mest) og sørge for at der er index på de felter som optræder i ON og WHERE.

Find en lærers klasser:

```
SELECT Klasse.Navn
FROM (Klasse JOIN LærerKlasse ON Klasse.Navn=LærerKlasse.Klassenavn) JOIN
     Lærer ON LærerKlasse.LærerID=Lærer.ID
WHERE Lærer.Navn = 'X';
```

Find en klasses lærer:

```
SELECT Lærer.Navn
FROM (Lærer JOIN LærerKlasse ON Lærer.ID=LærerKlasse.LærerID) JOIN
     Klasse ON LærerKlasse.Klassenavn=Klasse.Navn
WHERE Klasse.Navn = 'Y';
```

Find en elevs lærere:

```
SELECT Lærer.Navn
FROM ((Lærer JOIN LærerKlasse ON Lærer.ID=LærerKlasse.LærerID) JOIN
     Klasse ON LærerKlasse.Klassenavn=Klasse.Navn) JOIN
     Elev ON Klasse.Navn=Elev.Klassenavn
WHERE Elev.Navn = 'Z';
```

Vi konkluderer at vi vil have index på felterne:
- LærerKlasse.Klassenavn
- LærerKlasse.LærerID
- Lærer.Navn
- Elev.Navn

(der er allerede index på primær nøgler så dem springer vi over !)

Tabel struktur:

```
Elev
----
ID (PK) - INTEGER
Navn - VARCHAR(30) - Index
Alder - INTEGER
Klassenavn (FK) - VARCHAR(10)

Klasse
------
Navn (PK) - VARCHAR(10)
```

```
Lærer
-----
ID (PK) - INTEGER
Navn - VARCHAR(30) - Index

LærerKlasse
-----------
LærerID (delt PK,  FK) - INTEGER - Index
Klassenavn (delt PK, FK) - VARCHAR(10) - Index
```

Vi kan nu lave SQL til at lave databasen med:

```
CREATE TABLE Elev (
   ID INTEGER,
   Navn VARCHAR(30),
   Alder INTEGER,
   Klassenavn VARCHAR(10),
   PRIMARY KEY(ID)
);

CREATE TABLE Klasse (
   Navn VARCHAR(10),
   PRIMARY KEY(Navn)
);

CREATE TABLE Lærer (
   ID INTEGER,
   Navn VARCHAR(30),
   PRIMARY KEY(ID)
);

CREATE TABLE LærerKlasse (
  LærerID INTEGER,
  Klassenavn VARCHAR(10),
  PRIMARY KEY(LærerID,Klassenavn)
);

CREATE INDEX IndexElevNavn ON Elev(Navn);

CREATE INDEX IndexLærerNavn ON Lærer(Navn);

CREATE INDEX IndexLærerKlasseLærerID ON LærerKlasse(LærerID);

CREATE INDEX IndexLærerKlasseKlasseNavn ON LærerKlasse(KlasseNavn);
```

Så har vi nået målet !

Man bør nok iøvrigt undgå danske bogstaver i tabel og felt navne i praksis.

Eksemplet er naturligvis ret banalt. Men metodikken kan sagtens
anvendes på mere komplekse problemstillinger.

Kommentar af janbb d. 14. Jan 2004 | 1

Svaert emne. SQL er en top-ting

Kommentar af alvion d. 24. Mar 2004 | 2

God artikel. Jeg ville nu nok bytte rundt på punkt 4 og 7, da det så er lettere at finde ud af, hvilke felter der skal overføres som fremmmednøgler (det er jo altid primær nøglen, så den bør findes først).

Kommentar af simonvalter d. 14. Jan 2004 | 3

Rigtig godt forklaret, den database bog jeg har er 50 sider om at sige det samme, men de får vist også penge pr ord eller side, en artikel med normalisering ville være en god opfølgning ;)

Kommentar af hermandsen d. 16. Apr 2004 | 4

Udemærket gennemgang af tankerne bag opbygningen af en database... Dog forstod jeg ikke helt hvad fordelen var ved at lave index for nogle af tingene. Lidt mere info herom kunne være rart!! :)

Kommentar af pnr d. 08. Feb 2005 | 5

Super artikel, den giver en masse grundlæggende information, beskrevet kort og godt.

Kommentar af konder d. 04. Apr 2004 | 6

fin artikel -desværre forstod jeg ikke det med index på de mest naturlige queries

Kommentar af schoesler d. 24. Jan 2004 | 7

Kommentar af michaeltajo d. 14. Nov 2005 | 8

Pga. tidnød måtte jeg desværre nøjes med at skimme artiklen; men jeg vender helt sikkert tilbage...

Kommentar af mymouse d. 23. Mar 2004 | 9

Godt forklaret, kort og overskueligt.

Kommentar af trer d. 16. Feb 2004 | 10

Fint gået - en artikel man godt kan henvise begyndere til. Også god ide med historiken over ændringerne.

Kommentar af petkrug d. 25. Feb 2004 | 11

En ganske fornuftig artikel - som giver et hurtigt overblik over mulighederne. Kan dog virke lidt teksisk på en nybegynder som undertegnede.

Kommentar af rwj (nedlagt brugerprofil) d. 26. Feb 2004 | 12

Kommentar af gartzen (nedlagt brugerprofil) d. 11. Nov 2004 | 13

God artikel, men jeg kunne godt bruge noget mere info om hvad primary og foreign key er.

Kommentar af cebuano d. 06. Jan 2005 | 14

Super

Kommentar af Kgn1989 d. 19. Jun 2012 | 15

Fornem artikel Arne!

Den fik min "åbenbaring" frem omkring foreign keys og tabeller i tabeller kan man vel sige :)

Tak for god guide!