

Databaseprogramming

Database

(Db bruger / normalisering (SQL) / keys / relationship & constrain (SQL))

Teori oplæg:

Normalisering, Keys – Relationships og constraint

Normalisering

- **Atomic** (et styk data pr kolonne)
- INGEN **Partial Dependency**
 - Er alle "data" kolonne knyttet til en unik id?
- INGEN **Transitive Dependency**
 - Er der data som forsvinder hvis en række slettes?
- Værktøj til normalisering er Keys som skaber 'relationship' og 'constraints'
 - Primary/composite key
 - Foreign key
 - Prøv at overveje: Hvis man kan JOIN, hvorfor så bruge en foreign key?

• Keys

- Primary key
 - 1 ID kolonne
- composite key
 - 2 til flere kolonner som tilsammen defineres som ID
- Foreign key
 - Skaber "Relationship", men også "constraints"

Database øvelse 1/3

CRUD operationer mod database og tabeller

1. Med Management Studio, login på SQL Server som admin og med en SQL kommando udføre nedstående punkter, **alt sammen med SQL kommando'er som kan gemmes og gentages**:
 - 1.1 Opret en ny database med navn: **TEC**.
 - [SQL CREATE DATABASE Statement \(w3schools.com\)](https://www.w3schools.com/sql/sql_create_database.asp)
 - 1.2 Opret en ny tabel i TEC databasen med navn: **Student** med følgende kolonner:
Id (int, primary key og auto increment), **FirstName** (nvarchar), **LastName** (nvarchar),
Teacher (nvarchar), **Course** (nvarchar)
 - https://www.w3schools.com/sql/sql_autoincrement.asp
 - 1.3 Indset 4 "records" (rækker) i **Student** tabel :
{ "Martin", "Jensen", "Niels Olesen", "Grundlæggende programmering" }
{ "Patrik", "Nielsen", "Niels Olesen", "Database programmering" }
{ "Susanne", "Hansen", "Niels Olesen", "Studieteknik" }
{ "Thomas", "Olsen", "Peter Suni", "Clientside programmering" }
 - [SQL INSERT INTO Statement \(w3schools.com\)](https://www.w3schools.com/sql/sql_insert_into.asp)
2. Undersøg hvorledes det oprettede tabel **Student** kan normaliseres med følgende 3 normaliseringsregl (**Tips**: det skal ende med at blive til 4 tabeller!):
 - Atomic (ingen gentagende data, Id'er må gerne)
 - INGEN Partial Dependency (Løses via relationship og constraint)
 - INGEN - Transitive Dependency

Læse stof: relationship og constraint : https://www.w3schools.com/sql/sql_foreignkey.asp

Normalisering eksempel af Student tabel:

Students

Id	FirstName	LastName	Teacher	Course
1	Thomas	Nielsen	Niels Olesen	Databaseserver
2	Erik	Hansen	Niels Olesen	Databaseserver
3	Susanne	Jakobsen	Per Jensen	Webserver
4	Jonas	Elsborg	Ole Nielsen	Linuxserver

Students

Id	FirstName	LastName
1	Thomas	Nielsen
2	Erik	Jensen
3	Susanne	Jakobsen
4	Jonas	Elsborg

Teachers

Id	FirstName	LastName
1	Niels	Olesen
2	Per	Jensen
3	Ole	Nielsen

Course

Id	Name	TeacherId
1	Databaseserver	1
2	Webserver	2
4	Linuxserver	3

Funktionaliteten til at se hvilket elev har
hvilket fag passer nu bedre som en ny tabel:

Tilmelding

Id	StudentId	CourseId
1	1	1
2	1	2
3	1	3

fremmednøgler

Teori oplæg:

JOINS

Der findes følgende 4 join typer:

1. JOIN

- https://www.w3schools.com/sql/sql_join.asp

2. INNER JOIN

- https://www.w3schools.com/sql/sql_join_inner.asp

3. LEFT JOIN

- https://www.w3schools.com/sql/sql_join_left.asp


4. RIGHT JOIN

- https://www.w3schools.com/sql/sql_join_right.asp

Databaseøvelse 2/3

Queries

Design en (og kun en) **select** query som skal returnere alle elever sammen med deres fag samt lærer i følgende format:



	Elevens navn	Fag	Lærerens navn
1	Martin Jensen	Grundlæggende programmering	Niels Olesen
2	Martin Jensen	Database programmering	Niels Olesen
3	Martin Jensen	Studieteknik	Niels Olesen
4	Patrik Nielsen	Grundlæggende programmering	Niels Olesen
5	Patrik Nielsen	Database programmering	Niels Olesen
6	Patrik Nielsen	Studieteknik	Niels Olesen
7	Susanne Hansen	Grundlæggende programmering	Niels Olesen
8	Susanne Hansen	Database programmering	Niels Olesen
9	Susanne Hansen	Studieteknik	Niels Olesen
10	Thomas Olsen	Database programmering	Niels Olesen

Tips: Følgende teknologi bruges til denne øvelse: **join**, **concat()** og **Aliases**

- **Concat()**
 - https://www.w3schools.com/sql/func_sqlserver_concat.asp
- **Aliases**
 - https://www.w3schools.com/sql/sql_alias.asp

Database øvelse 3/3

Login bruger vs. Database bruger

Opretter man en database har man også en bruger, en **database bruger**.

Database bruger

- **Database bruger** **ER IKKE** det samme som **Login bruger**.
 - Login bruger
 - Login på SQL server med rettigheder rettet mod selve serveren.
 - Database bruger
 - Rettigheder rettet mod specifikke database.

Opgave

1. Bruger: **Niels** skal arbejde med denne **TEC** database.
 - Opdater bruger **Niels** så han har alle rettigheder mod **TEC** databasen (kan opret, opdater og slet tabeller samt TEC database).
2. En anden bruger (Henrik) skal kun lave queries på databasen.
 - Opret en bruger: **Henrik**, som kun har læse rettighed.