# Database 04

|  |  |
| --- | --- |
| **Læringsmål** | Du kan:   * **1Pf4:** anvende centrale metoder og teknikker til at realisere modeller i et databasesystem […] |
| **Forventet læringsudbytte [SOLO]** | Programmeringssporet:   * [Unistrukturel] Du kan nævne flere kendetegn ved **database design** * [Unistrukturel] Du kan beskrive flere kendetegn ved en tabel (**database implementering**) |
| **Din forberedelse** | Programmeringssporet:   * Læs Why Use A Database.pdf (kan findes på ItsLearning)   + Side 20-27, dvs. hen til (men ekskl.) “Processing Relational Tables” * Se video omkring [Basic SQL](https://www.linkedin.com/learning/programming-foundations-databases-2/basic-sql) [video: 3:54]   + Med fokus på termerne ’SQL-statement’, ’DML’, ’Clause’, ’Expression’ og ’Query’ * I relation til [Constraints](https://www.w3schools.com/sql/sql_constraints.asp) (Med fokus på SQL Server)   + [Primary Key](https://www.w3schools.com/sql/sql_primarykey.asp)   + [Foreign Key](https://www.w3schools.com/sql/sql_foreignkey.asp)   + [Not null](https://www.w3schools.com/sql/sql_notnull.asp) * Genlæs / gense forberedelsen fra uge 6 (Ex39 og Ex40), herunder med fokus på:   + Hele den praktiske guide til design og implementering af relationelle SQL-databaser (Kom i gang med databaser.docx på ItsLearning)   + SQL-sætningerne (fra forberedelsen til Ex40) |

# Quote of the day

“In God we trust. All others must bring data.”

[*W. Edwards Deming*](https://www.deming.org/deming/deming-the-man)*, statistician, professor, author, lecturer, and consultant.*

# Øvelse 1: Terminologi

Del teamet op i to mindre grupper og brug **Ordet rundt** til at reflektere over begreberne: ’SQL-statement’, ’SQL-betingelse (en: SQL Clause)’, ’DML’, ’Expression’ og ’Forespørgsel (en: Query)’. Sørg for at alle får mulighed for at tale.

*Tidsramme: 20 minutter*

# Øvelse 2: Pet Paradise

Dyreklinikken **Pet Paradise** har mere eller mindre desperat henvendt sig til dig for at få hjælp til deres mest centrale arbejdsgang. Klinikken anvender et regneark som omdrejningspunkt for registrering af alle deres kunder (både dyr og ejere) samt evt. udførte behandlinger. Efter et par ’uheld’ med regnearket, er klinikken godt træt af kun at have dette ene regneark, som alle medarbejdere skal opdatere (mange gange samtidigt), når der skal registreres kunder og behandlinger. Du skal lave en mere holdbar løsning til dem, så klinikken ikke ender med at være Pet Hell i stedet for.

Du får udleveret et regneark med data fra dyreklinikken og skal ud fra dette regneark udvikle en fleksibel databaseløsning for dem.

I de efterfølgende øvelser skal du ud fra regnearket udarbejde en domænemodel og herefter skal du gennem to ud af de tre faser i udvikling og anvendelse af databaser, som beskrevet i ”Kom godt i gang med databaser”, mere specifikt skal du se på database-design og database-implementering (SQL).

Du skal også realisere flere realistiske forespørgsler til databasen, så du er sikker på, at din løsning fungerer efter hensigten.

Benyt den CL-struktur, du finder bedst egnet til de følgende øvelser.

## Øvelse 2.1: Problemer med regnearket

Pet Paradise kan ikke altid forstå hvilke problemer, regnearket giver, kun at det ofte går galt, når flere medarbejdere skal bruge det. Du skal nu identificere potentielle problemer med at vedligeholde regnearket, så du kan forklare det til klinikken og foreslå en bedre løsning.

Det udleverede regneark ”Ex42-Pet Data.xlsx” kan hentes i Git materiale-folderen (Benyt git pull).

* Analysér regnearket og find de potentielle udfordringer (*tænk især på forberedelsen angående ”Why use a database” i denne forbindelse*)

*Bemærk, at enheden på ’PetWeight’-kolonnen er pund, og enheden på Charge-kolonnen er US dollars.*

## Øvelse 2.2: Fra regneark til domænemodel

Gennemse regnearket og dan dernæst en domænemodel ud fra dets data. Udfør følgende:

* Overvej hvilke kolonner i regnearket hører sammen, dvs. har samme tema, og dan en domæne­klasse for hver af disse temaer. Fastlæg også relationerne inkl. kardinalitet (multiplicitet) mellem domæneklasserne
* Tegn domænemodellen i et passende grafisk værktøj ([UMLet](https://www.umlet.com/) eller lign.)

## Øvelse 2.3: Fra domænemodel til relationelt databaseskema

Du skal nu udvikle et relationelt databaseskema ud fra domænemodellen, hvor du anvender **tekstnotationen** til at angive tabellerne samt deres felter, primære nøgler og fremmednøgler.

Benyt fremgangsmåden angivet i DatabaseDesign-dokumentet og udfør følgende:

* Identificér alle tabeller nødvendige til at repræsentere din DCD fra forrige øvelse
* For hver tabel:
  + beskriv alle dets attributter
  + Angiv primærnøgler og fremmednøgler til at repræsentere alle relationer mellem tabeller

## Øvelse 2.4: Fra domænemodel til DCD

Du skal nu specificere en DCD for din domænemodel, hvor du specielt overvejer hvilke datatyper de enkelte attributter skal have og evt. navngivning.

Tegn DCD’et i et passende grafisk værktøj.

## Øvelse 2.5: Fra relationelt databaseskema til UML-database model

Næste trin er at udarbejde et relationelt databaseskema, som grafisk illustrerer alle tabeller, felter, nøgler samt relationer mellem tabeller.

Du skal anvende UML til at specificere modellen med. Brug UML notationen som angivet i forberedelsen. Tegn databasemodellen grafisk i et passende værktøj.

## Øvelse 2.6: Fra UML-database model til SQL-sætninger

Du skal nu danne SQL-sætninger til at oprette alle dine tabeller i din UML-database model samt indsætte data i dem. Vær opmærksom på, at det ikke er alle felter i regnearket, der indeholder data. Det skal databasen kunne håndtere.

Udfør følgende:

* Dan alle SQL-sætninger (CREATE TABLE) til at oprette tabellerne og deres indbyrdes sammenhænge (PK, FK constraints)
* Dan dernæst alle SQL-sætninger (INSERT INTO) til at indsætte regnearkets data i dem.

**Anbefaling:** Gem alle SQL-sætninger i en .sql-fil på GitLab / GitHub (hvad I nu bruger i teamet).

## Øvelse 2.7: Fra SQL-sætninger til realisering i SQL Server

Via SQL Server Management Studio eksekvér de SQL-sætninger, du dannede i forrige øvelse til at oprette dine tabeller og data i databasen. Nu er databasen med indhold klar til brug.

## Øvelse 2.8: Undersøg og afprøv, hvordan SELECT og constraints virker

Gå ind på sqlbolt.com, benyt deres interaktive vejledning (en: interactive tutorial).

Læs og udfør de tre første øvelser (en: lessons):

* [SQL Lesson 1: SELECT queries 101](https://sqlbolt.com/lesson/select_queries_introduction)
* [SQL Lesson 2: Queries with constraints (Pt. 1)](https://sqlbolt.com/lesson/select_queries_with_constraints)
* [SQL Lesson 3: Queries with constraints (Pt. 2)](https://sqlbolt.com/lesson/select_queries_with_constraints_pt_2)

## Øvelse 2.9: Pet Paradise forespørgsler

Du har nu lavet en fungerende databaseløsning til Pet Paradise og skal nu udføre forskellige simple forespørgsler med SELECT-kommandoen i databasen for at sikre dig, at din løsning fungerer. Du skal også benytte dig af constraints i denne forbindelse.

Udfør følgende forespørgsler:

* Find og vis ’OwnerFirstName’, ’OwnerLastName’ og ’OwnerEmail’ for alle kæledyrsejere (Owners), som har en NULL-værdi i ’OwnerPhone’
* Find antal kæledyr i Pet-tabellen
* Find og vis ’PetName’, ’PetBreed’, ’PetType’ og ’PetDOB’ for alle kæledyr med ’PetType’ sat til ’Dog’ og ’PetBreed’ sat til ’Std. Poodle’
* Find og vis alle hunde (alle kolonner) i Pet-tabellen, som vejer mindst 20 pund (PetWeight).
* Find alle behandlinger (alle kolonner), som er udført i tidsrummet 1. september 2014 til 30. september 2014