

**Wiki****Dokumentation • Database • C# • Tests • Log, Tidsplan • Manual**

# Dokumentation

## Analyse

Opgaveformuleringen: Krav specifikationen fra Sydvest-Bo fortæller om nogle ret konkrete relationer som eksister imellem de informationer, data firmaet håndterer:

- En række personer
- der optræder i forskellige sammenhænge
- og udfylder forskellige roller og gøremål, resulterende i at der produceres
- 4 konkrete kontrakt typer
- og noget kommunikation
- og Noget bruger interaktivitet
- og lidt restriktioner.

Vores Database design kæder relationerne sammen på en konsistent måde, men det efterlader ret åbenlyse huller. Hertil introducerer vi nogle views i databasen:

- En udlejningstabel, hvoraf det fremgår hvornår noget er ledigt
- Opgave lister til medarbejdere

Der stilles også krav om .... [ /#&%/(% ]

De enkelte konsulenter, kunder, udlejere og handy-mænd skal på forskellig vis kunne navigere rundt og betjene programmet, så de intuitivt får nogle relevante informationer og andre måske hemmelig holdes.

Derfor er det ønskelig med en GUI der kan levere dette og f.eks. kan identificere de enkelte brugere vha. et login.

## ER diagram, roller, opgaver og funktioner

### Distrikt

#### Egenskaber og Relationer

- Et distrikt har et område navn
- Et distrikt kan have mange Ferieboliger
- Et distrikt kan have mange Opsynsmænd
- Et distrikt har en Udlejningskonsulent (burde have mange)
- Et distrikt har ikke en slags adresse, ikke en geografisk udstrækning i form af post numre.

## Opgaver

- Finde ud af hvilke folk det har tilknyttet.
- Finde ud af hvilke ferieboliger det har tilknyttet.

## Program Funktioner

- ingen login
- Udlejningskonsulenten skal kunne:
  - administrere områder ?

## Feriebolig

Vi har lagt Sommerhuse og lejligheds komplekser sammen til Feriebolig. Derfor optræder der en egenskab kaldet FerieboligType

## Egenskaber og Relationer

- Størrelse
- Rum (antal)
- Senge (antal)
- Kvalitet - Udlejningskonsulentens blåstempel. Måske skala 0-10.
- Pris - en base pris fra hvilken man kan udregne noget sæson tillæg og eventuel tilbuds rabat.
- FerieboligType
- m.m. kan anføres under Noter

## Program Funktioner

- Udlejningskonsulent skal kunne
  - Oprette Feriebolig
  - Rette Feriebolig
  - Slette Feriebolig
- Kundekonsulenten skal kunne:
  - Oprette en reservation /lejekontrakt
  - Rette en reservation /lejekontrakt
  - Slette en reservation /lejekontrakt

## Opgaver

## Udlejningskontrakt

Udlejningskontrakten er relationen imellem feriebolig, dens ejer, samt udlejningskonsulenten. Det er for såvidt godt nok, men hvis der ikke er nogen udlejning, er der heller ikke nogen relation, og man kan således ikke finde ejeren af et sommerhus. Det kan løses på to måder:

1. Kontrakten oprettes under alle omstændigheder og har en liste af udlejninger der godt kan være tom.
2. Kontrakterne oprettes når løbende pr udlejning. Men til start oprettes en bagud dateret 0 kontrakt der fastslår relationen.

## Egenskaber og Relationer

- Udlejnings Aar
- Udlejnings Uge

- Pris

## Opgaver

# Lejekontrakt

## Opgaver

### Program Funktioner

- login

# Udlejningskonsulent

### Egenskaber og Relationer

- Er tilknyttet et distrikt
- Tager sig af kommunikation med Feriebolig ejere
- Klassificere (og godkende) Ferieboliger (1-10)
- Ansvar for udlejningskontrakter, (Sudvest-Bo - Ejer) og (Sudvest-Bo - inspektør)
- Engagere Opsynsmænd

## Opgaver

- Administrere Sæsonkategorier

### Program Funktioner

- login
- Udlejningskonsulent skal kunne
  - Administrere Lejligheder:  
Dvs få en særskilt liste af dem Sommerhus og Lejligheder
  - Opret Feriebolig
  - Rette Feriebolig
  - Slette Feriebolig
- Udlejningskonsulent skal kunne
  - Administrere Lejligheds Inspektører:  
Dvs få en særskilt liste af dem Sommerhus ejere og Lejligheds Inspektører
- Udlejningskonsulent skal kunne
  - Opret Ejer
  - Rette Ejer
  - Slette Ejer
- Udlejningskonsulenten skal kunne:
  - administrere områder ?
- Udlejningskonsulenter og Kundekonsulenter skal kunne:
  - Administrere Sæsonkategorier

# Kundekonsulent

### Egenskaber og Relationer

## Opgaver

Klassificere ferieboliger

### Program Funktioner

- login
- Kundekonsulenten skal kunne:
  - Få listet hver ferieboligs reservationer /lejekontrakter.  
herigennem se en specifikation af hver enkelt udlejning.
  - Oprette en reservation /lejekontrakt
  - Rette en reservation /lejekontrakt
  - Slette en reservation /lejekontrakt
- Udlejningskonsulenter og Kundekonsulenter skal kunne:
  - Administrere Sæsonkategorier.

## Opsynsmænd

Opsynsmænd virker på sommerhuse, denne rolle indtages af Lejligheds inspektorerne.

### Egenskaber og Relationer

- Timebetaling
- Aflæsning af Strømforbrug
- Eftersyn
- Lørdags opgaver
- kalender / ugeplan , hvilke ferieboliger skal chekkes.

## Opgaver

### Program Funktioner

- login

## Ejer

### Egenskaber og Relationer

- Leje en feriebolig
- modtage en lejekontrakt
- betale

## Opgaver

### Program Funktioner

- Ejer Skal kunne
  - login
  - Se hvilke boliger vedkommende har registreret
  - Se hvilke leje kontrakter vedkommende har
  - Eventuelt se hvor når hvad er ledigt

- Udlejningskonsulent skal kunne
  - Opret Ejer
  - Rette Ejer
  - Slette Ejer

## Kunder

### Egenskaber og Relationer

- Person data
- Adresse data
- Password

### Opgaver

- kontakte en udlejnings konsulent og leje en feriebolig
- kontakte en udlejnings konsulent og kunne rette en lejekontrakt for en feriebolig

### Program Funktioner

- login
- Kunder skal kunne en liste af deres egne lejekontrakter
  - herunder se strømforbrug
  - betale for lejekontrakten
  - betale for strømforbrug efterfølgende
  - Få en status over hvad der er betalt.
- Kunder skal kunne få en liste af deres egne person data
- Kunder skal kunne få kontakt oplysninger deres Udlejnings- og Kunde-konsulenter, tilknyttet hver lejemål

## Sæsonkategori

Der er 4 sæsoner:

### Egenskaber og Relationer

- Kategori 1: Lav: vinter
- Kategori 2: Mellem: sommerhalvår
- Kategori 3: Høj: Ferie: Nytårs uge, Vinterferie, påske, sommerferie ( dansk og udenlandsk), efterårsferie
- Kategori 4: Super: Uge: 28, 29, 30, 52
- Skal have en relateret prismodifikator. En faktor basePrisen kan modificeres med.

### Opgaver

### Program Funktioner

- Udlejningskonsulenter og Kundekonsulenter skal kunne:
  - Administrere Sæsonkategorier.  
Dvs. rette i de enkelte kategories perioders uger.  
Måske skal sæson kategorierne også kunne udvides og deres navne tilrettes.  
Men det skal ikke kunne være mulig at efterlade en uge uden at der er tilknytning til en kategori.

# Person

## Egenskaber og Relationer

- Person data:
  - Personid
  - Fornavn
  - Efternavn
  - Tlf
  - Email
  - Password
  - FK Adresseid

## Opgaver

### Program Funktioner

- login  
Selve funktionen skal bo under persontypen: Kunde, Udlejningskonsulent m.f. men brugernavn (email) og passwordet ligge i denne tabel
- En person skal kunder kunne:
  - Konsulenter skal kunne oprette en person.
  - Konsulenter skal kunne rette i person data.
  - ? ? Skal nogen kunne slette en person ? ?  
Det tror jeg egentlig ikke - i hvert tilfælde ikke en kunde, før denne betalt og opgjort moms og skat.
  - ? ? Måske skal et blankt password betyde at kunden ikke kan logge ind ? ?
  - Udskrive person data: (måske på nær password)
  - Returnere person data, til andre objekter: (på nær password)
  - Rette password
  - Returnere Login Brugernavn og password til login funktionen for validering
- Kunder skal kunne få en liste af deres egne person data
- Kunder skal kunne få kontakt oplysninger deres Udlejnings- og Kunde-konsulenter, tilknyttet hver lejemål

# Adresse

## Egenskaber og Relationer

- adresse data:
  - Adresseid
  - Adresse
  - FK Postnr
- Personer og Ferieboliger skal kunne have en adresse

## Opgaver

### Program Funktioner

- En adresse skal kunder kunne:
  - Skal kunne oprettes.
  - Skal kunne rettes.
  - Skal kunne slettes hvis der ikke er nogen/noget der har en relation til den.

- Konsulenter skal kunne rette i person data.
- Udskrive Adresse, med postnr og bynavn.
- Returnere Adresse, med postnr og bynavn, til andre objekter.

## PostnrBy

### Egenskaber og Relationer

- PostnrBy data:
  - PK Postnr.
  - Bynavn
- Ligger i en adresse

### Opgaver

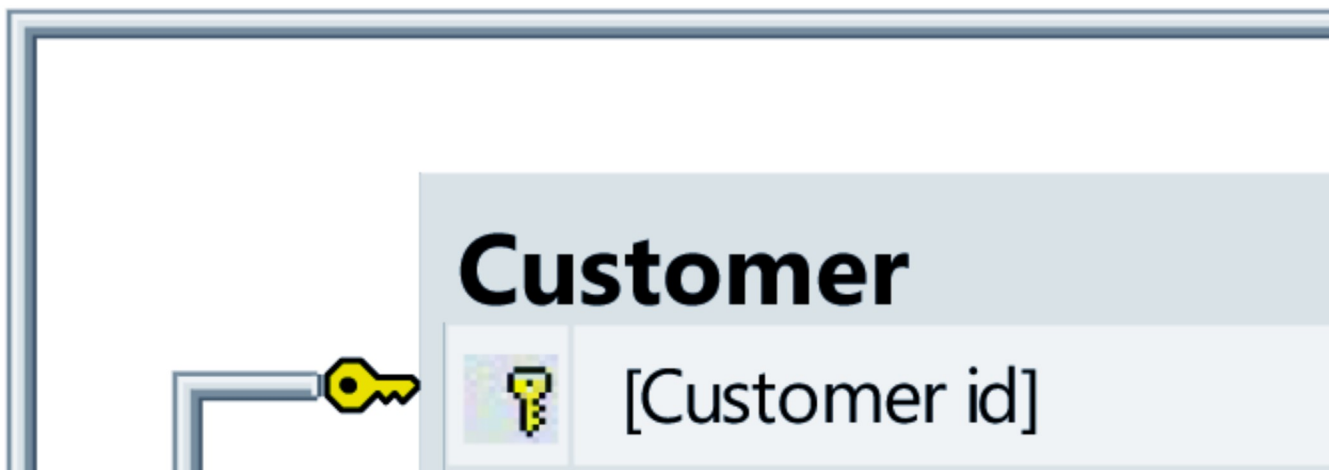
### Program Funktioner

- Et Postnummer skal kunder kunne:
  - Udskrive Postnr og Bynavn
  - Returnere Postnr og Bynavn til Adresse objektet.
  - Skal ikke kunne oprettes.
  - Skal ikke kunne rettes.
  - Skal ikke kunne slettes.
  - Skal administreres af en Database administrator.

## DB


Vores database (db) design afspejler sammenhængene skitseret ovenfor. Der er lagt vægt på at opdele tabellerne i forhold til de relationer der optræder i kravspecifikationen og der er gjort et forsøg på at udskille redundante data i mindre tabeller. Intuitive sammenkog fra databasen er lagt over views, og unintuitive sammenhænge er lagt over i metoder der laver mere komplekse forespørgsler til databasen. Dette skyldes at disse metoder skal kunne arbejde på en opdaterede data fra basen, og ikke et view der blev auto genereret tidligere. Det kunne f.eks. være at en udlejning netop har fundet sted og at en anden kunde samtidig prøver at leje den samme hybel samtidig. Et en ad gangen tjek, efter først til mølle princippet.

Krav specifikationen rummer mulighed for en rationalisering af database designet ved at lade alle ferieresidenser, deres ejere og opsynsmænd bo i fælles klasser. Disse respektive tabeller tilføjes et type felt til arts bestemmelse.



	[Person id]
	Notes

## Customer Consulta

	[Customer Consultant i
	[Person id]
	Etc




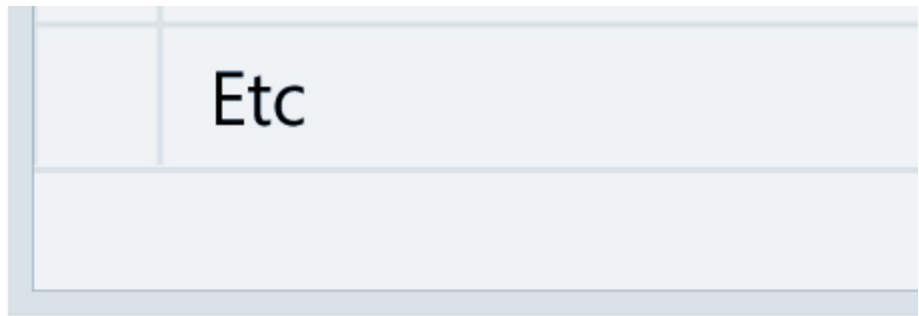
## Customer Rental C

	[Customer id]
--	---------------



	[Customer Consultant id]
	[Rental Consultant id]
	[Date of contract]
	[Date Start Rent]
	[Date End Rent]
	Price
	[Price modifier]
	[Customer Rental Contr

<b>Residence Owner</b>	
	[Residence Owner id]
	[Person id]



Det eneste der egentlige stikker ud, er introduktionen af et password. Denne attribut der sidder på enhver person. Af flere årsager, men bl.a. som udslag af et køns-kompromis, har hver person altså, uanset køn, netop en disse attributter.

Af tabellernes design følger en række intuitive klasser.

## Intuitive Klasser

De intuitive klasser udgøres primært som en parallelitet til databasens tabeller. Ved at sammensættes og arve fra hinanden og skal disse tabel-klasser kunne opbygge, rumme, opdatere og slette data fra databasen. De skal rumme metoder til at sammenstrikke hele sql sætninger eller i det mindste få jobbet udført.

Parallelt skal disse klasser kunne indgå i et lignende regi til at afgive bruger I/O: indhold til skærmen, en kontrakt, en email. Overordnet skal output fra nedarvede klasser og metoder kunne returnere små formaterede strenge. Disse skal igen af en overordnet klasse kunne sammenflettes til linjer af formateret tekst, der igen kan afleveres til noget: GUI, email eller db'en, som håndtere og laver det endelige output format.

## Gui

For at udnytte det maximale af konsolskærmen benytter vi et par biblioteker til at maksimere skærmens areal:

```
using System.Runtime.InteropServices;
using System.Text;
using System.IO;
```

Herved få vi også fastlagt skærmen størrelse på brugerens computer og kan fastslå de eksakte koordinaters og mål til at lave brugerfladen: et centrum, en ønskelig pop-op vindues størrelse og lokation, placeringer af et vindues heiraki/stak, proportionerne imellem vinduesbufferen og skærmstørrelsen, mm.

## Objekt vindue

Vi laver et objekt pop-op-vindue der kan drage nytte af disse egenskaber. Objekt Property x og y fortæller hvor vinduet starter: ca. centrum -1/2 skærm udstrækning og lidt fra... Objekt Property w og h (width og height) er udstrækningen af pop-op-vinduet: ca. 2 \* (centrum -1/2 skærm udstrækning og lidt fra).

### Graphic Layout

(x,y),z	center	(x+w,y)
---------	--------	---------

	En margin definere hvor indholdet starter. (x-mw,y-mh)	
	Dette felt kan altså slettes, farves med baggrundsfarve og space	
(x,y+h)	center	(x+w,y+h)

Vi kan ombryde tekst i vinduet, da vi nu kender vinduets udstrækningen. Varierede metoder laves til at udskrive tekstbeskeder, menuer, Tabel indhold, placering af tekst input.

Dette fordre nogle igen forskellige typer af vinduer: menuer, pop-op, tekst input, scrolle til sager der ikke kan være på skærmen. (Her satses benhårdt på at de uattraktive ferieuger ikke automatisk er i fokus) Når der trækkes lidt fra af hensyn li layoutet, marginer, skygger og placering på skærmen.

## Object Window design

Object window	
int x	(x,y) koordinatet
int y	Definerer hvor vinduets Venstre Top hjørne ligger.
int w	(x+w,y+h) koordinatet (width and heigth)
int h	Definerer hvor vinduets Højre Bund hjørne ligger.
int z	Z index ligger forsøgsvis her, kunne også være en id.
bool visible	Skal vinduex vises eller ej

## Objekt skærm

Uden om vinduet kan vi bygge et objekt skærm, der holder en stak eller liste af vindues instanser. Elementerne kan tilføjes, byttes rundt eller fjernes. Gennem løbes stakken tegnes gentegnes skærmen i den ønskede rækkefølge.

## Objekt skærmrække

Ved at pakke objekt skærme ind i et nyt objekt som ovenfor, kan vi facilitere at flere skærme koeksistere men kun en enkelt vises ad gangen.

## Mindre intuitive klasser og metoder