### AestasGIS



NOTAT Vedr. Ballerup PArk system

ver. 1.1

13/6 2024

### Bo Victor Thomsen

# Opsummering

Ballerup kommune ønsker at konvertere deres system til administration af grønne områder fra SWECO Park system til lokal placeret database system baseret på PostgreSQL/PostGIS; QGIS til behandling og redigering af data samt MapCentia GC2 til præsentation af kort og data via et Web baseret brugerflade.

Dette notat omhandler struktur i den nye database. Fremtidige tilføjelser til notatet vil indeholde funktionskrav til og -beskrivelser af det nye system.   
  
Selve teksten er en første sammenskrivning af et sæt løse noter (”gule lapper”) og er derfor ikke særlig velskrevet.

Database struktur er fastlagt ved analyse af den eksisterende SWECO-struktur samt empirisk viden fra tilsvarende systemer.

# Systemets hovedstruktur

Data i det nye system opdeles i 6 grundlæggende datablokke: Elementer, Områder, Tilsyn, Kontakter, Prissætninger og Aktiviteter. Der er en række datatype/begreber, som er fælles for flere datablokke i systemet.

# Elementer

Elementer er den grundlæggende geometriske data-del af systemet. Den indeholder tre tabeller til lagring af hhv. punkter, linjer og flader.

Elementerne er defineret ved deres typer, som bestemmer hvad geometrierne repræsenterer: Veje, grusstier, græsplæner, buske osv.   
  
Der findes to tabeller til typedefinition af elementerne: *element\_typer*, f.eks. ”Grussti”, ”3m-vej”, ”Prydplæne”, ”Sportsplæne”, ”Asfaltbelægnign” osv. : Element typerne er opdelt i grupper / overordnede typer: *elementgruppe\_typer*, f.eks. ”Græs” , ”Veje”, ”Belægning”.  
   
Erfaringsmæssigt har det vist sig, at et todelt type-hierarki er tilstrækkeligt. Om nødvendigt kan dette findeles yderligere.

Hver element tabel indeholder information om element-typen samt div. standardoplysninger. såsom areal, omkreds for flader; længde for linjer osv.

Ekstra information til elementer som afhænger af elementets specifikke type, placeres i én kolonne som kan indeholde en JSON struktur kaldet ”ekstra”. Denne JSON stuktur kan inholde enhver form for alfanumerisk information.   
  
Den valgte metode giver en stærkt forenklet database struktur i forhold til andre systemer: Alle data til et element er placeret i *én* post i *én* tabel. Til visningsforhold kan indholdet af JSON feltet ”foldes ud” via views i databasen, således at data elementer fra JSON feltet optræder som regulære kolonner i viewet.   
  
NB! Der mangler en metodebeskrivelse for at kvalitetssikre disse JSON-data

Element-tabellerne, inkl. typedefinitions-tabeller er placeret i schema ”elementer”   
Denne datablok er strukturmæssigt forbundet til resten af databasen via en række tabeller, som beskrives senere i rapporten.

# Områder

Områder er en geometrisk opdeling af kommunen. De findes tre typer område geometrier:

* ”Hele kommunen”, som er et polygon der dækker hele kommunen (!). Bruges til aktiviteter, som *ikke* skal splittes op i mindre områder.
* ”Aktivitetsområder”. Områder, som dækker mindre eller større dele af kommunen. Disse områder kan godt overlappe hinanden og de øvrige områder. Benyttes oftest som en områdebegrænser for aktivitetsgrupper (se senere i notat)
* ”Administrative delområder”. Disse områder danner en fladedækkende inddeling af hele kommunen, dvs. kombinationen af samtlige disse områdepolygoner dækker det samme område som kommunepolygonen. Alle delområde polygon støder op til hinanden, men har intet overlap

De tre typer af områdepolygoner benyttes til bruges til opdeling af geometrier i blok Elementer samt semiautomatisk tildeling af aktiviteter til elementer via tilhørsforhold for elmenter baseret på aktivitetsområder og element-typer  
  
Aktivitetsområder er endvidere forbundet til aktivitets-grupper via en relation. Aktivitetsgrupper og deres relation til hhv. områder, elementer og entrepriser beskrives senere.

Der findes endvidere en lookup tabel for-områdetyper. Både tabel områder og lookup tabel er placeret i schema ”omraader”

# Tilsyn

Tilsyn bliver et sæt af tabeller og funktioner til at understøtte tilsyns-funktioner for kommunens grønen og grå elementer.

Det forventes, at denne del skal fungere sammen med Android/IPad baserede feltsystemer, såsom ”Mergin” eller ”QField”. Der er pt. ingen nærmere definition, da opsætning af denne del fuldstændig afhænger af funktionalitetsønsker, valg af udstyr og felt programmer.

# Kontakter

Kontakter er en enkelt tabel, som indeholder adresse og kontaktoplysninger for aktører: Firmaer, institutioner, Medarbejdere i kommunen osv..  
  
Kontakter benyttes som referencer i hhv. entrepriser og aktivitetsgrupper.

Tabellen befinder sig i schema kontakter.

# Prissætninger

Prissætninger er en række tabeller, som definerer prisen på enkelt-aktiviteter.

Prissætning deles op i ”Entrepriser”, En entreprise er defineret ved en liste af referencer til aktivitets-grupper, som hver definerer et sæt aktiviteter der foretages på elementer i geografisk område, en entreprenør (hvem, der udfører opgaverne – en reference til en post i ”kontakt” tabellen) samt en prisliste over potentielle enkelt- aktiviteter som udføres indenfor entreprisen. Hver aktivitet i prislisten inkluder en enhedspris for aktiviteten. Denne metode til prissætning kræver, at man faktisk prissætter en given entreprise som summen af alle aktiviter gange med deres respektive enhedspris. Det kan være, at metodevakget er for restriktivt.

Hvis der er flere entreprenører, findes kan der være forskellige enhedspriser for samme enkelt-aktivitet. En entreprenør kan også være kommunens egne afdelinger.

Der findes endvidere reference tabeller som definerer hvilke aktivitetsgrupper (Se næste afsnit) der udføres af den en enkelte entreprenør.

# Aktiviteter

Aktiviteter er den datablok, som definerer hvilke aktiviteter, der udføres på de enkelte elementer. Begrebet er delt op i flere niveauer:

* Aktivitetsgrupper - Hver aktivitetsgruppe kan opfattes som en del-entreprise på et bestemt område defineret ved en reference til en post i område-tabellen. Plus en reference til én specifik entreprise i prissætningsdelen.
* Aktivitetstyper - Denne tabel indeholder en liste over samtlige aktiviteter, som kan foretages, f.eks. klipning af hække, slåning af græs, vintervedligeholdelse af veje, tømning af kommunen offentlige skraldespande osv. osv. Aktivitetstyperne er
* generel for alle aktivitetsgrupper.   
    
  Den enkelte post i denne type indeholder dels en beskrivelse af aktiviteten samt en reference til en elementgruppe. Denne elementgruppe-reference bruges som et filter, således muligheden for at definere irrelevante aktiviteter til de forskellige elementer fjernes (Græsplæner kan ikke ryddes for sne, Veje kan ikke hækklippes osv.)
* Element-aktivitet: Denne tabel definer sammenhængen mellem enkelt elementer (punkter, linjer, flader – hver med deres element-type) og en given aktivitet. Endvidere findes en reference til hvilken aktivitetsgruppe element-aktiviteten tilhører.

# Bilag A, E/R diagram

## A screenshot of a computer Description automatically generated

# Bilag B, oprettelse af database i SQL

CREATE SCHEMA elementer;  
CREATE SCHEMA kontakter;  
CREATE SCHEMA tilsyn;  
CREATE SCHEMA prissaetning;  
CREATE SCHEMA aktiviteter;  
CREATE SCHEMA omraader;

-- Table: element\_gruppetyper

CREATE TABLE elementer.elementgruppe\_typer (  
 gruppe\_navn character varying NOT NULL,  
 basis\_type character varying(1) NOT NULL,  
 passiv boolean NOT NULL DEFAULT false,  
 beskrivelse character varying,  
 CONSTRAINT element\_gruppetyper\_pk PRIMARY KEY (gruppe\_navn,basis\_type)  
);

-- Table: element\_typer

CREATE TABLE elementer.element\_typer (  
 gruppe\_navn character varying NOT NULL,  
 type\_navn character varying NOT NULL,  
 basis\_type character varying(1) NOT NULL,  
 passiv boolean NOT NULL DEFAULT false,  
 beskrivelse character varying,  
 CONSTRAINT element\_typer\_pk PRIMARY KEY (gruppe\_navn,type\_navn,basis\_type)  
);

ALTER TABLE elementer.element\_typer ADD CONSTRAINT element\_typer\_elementgruppe\_typer\_fk  
 FOREIGN KEY (gruppe\_navn, basis\_type)  
 REFERENCES elementer.elementgruppe\_typer (gruppe\_navn, basis\_type);

-- Table: elementer\_flader

CREATE TABLE elementer.elementer\_flader (  
 id uuid NOT NULL DEFAULT gen\_random\_uuid(),  
 basis\_type character varying(1) NOT NULL DEFAULT 'f'::character,  
 element\_gruppe character varying NOT NULL,  
 element\_type character varying NOT NULL,  
 areal double precision,  
 omkreds double precision,  
 hoejde double precision,  
 bredde double precision,  
 ekstra jsonb,  
 geom public.geometry(MultiPolygon,25832) NOT NULL,  
 CONSTRAINT elementer\_flader\_pk PRIMARY KEY (id)  
);

ALTER TABLE elementer.elementer\_flader ADD CONSTRAINT elementer\_flader\_element\_typer\_fk  
 FOREIGN KEY (element\_gruppe, element\_type, basis\_type)  
 REFERENCES elementer.element\_typer (gruppe\_navn, type\_navn, basis\_type);

-- Table: elementer\_linjer

CREATE TABLE elementer.elementer\_linjer (  
 id uuid NOT NULL DEFAULT gen\_random\_uuid(),  
 basis\_type character varying(1) NOT NULL DEFAULT 'l'::character,  
 element\_gruppe character varying NOT NULL,  
 element\_type character varying NOT NULL,  
 laengde double precision,  
 hoejde double precision,  
 bredde double precision,  
 ekstra jsonb,  
 geom public.geometry(MultiLineString,25832) NOT NULL,  
 CONSTRAINT elementer\_linjer\_pk PRIMARY KEY (id)  
);

ALTER TABLE elementer.elementer\_linjer ADD CONSTRAINT elementer\_linjer\_element\_typer\_fk  
 FOREIGN KEY (element\_gruppe, element\_type, basis\_type)  
 REFERENCES elementer.element\_typer (gruppe\_navn, type\_navn, basis\_type);

-- Table: elementer\_punkter  
CREATE TABLE elementer.elementer\_punkter (  
 id uuid NOT NULL DEFAULT gen\_random\_uuid(),  
 basis\_type character varying(1) NOT NULL DEFAULT 'p'::character,  
 element\_gruppe character varying NOT NULL,  
 element\_type character varying NOT NULL,  
 hoejde double precision,  
 bredde double precision,  
 ekstra jsonb,  
 geom public.geometry(Point,25832) NOT NULL,  
 CONSTRAINT elementer\_punkter\_pk PRIMARY KEY (id)  
);

ALTER TABLE elementer.elementer\_punkter ADD CONSTRAINT elementer\_punkter\_element\_typer\_fk  
 FOREIGN KEY (element\_gruppe, element\_type, basis\_type)  
 REFERENCES elementer.element\_typer (gruppe\_navn, type\_navn, basis\_type);

-- Table: kontakter  
CREATE TABLE kontakter.kontakter (  
 id uuid NOT NULL DEFAULT gen\_random\_uuid(),  
 navn character varying NOT NULL,  
 adresse character varying,  
 postnr character varying,  
 postby character varying,  
 telefon\_nummer character varying,  
 mail\_adresse character varying,  
 passiv boolean NOT NULL DEFAULT false,  
 CONSTRAINT kontakter\_pk PRIMARY KEY (id)  
);

-- Table: omraade\_typer

CREATE TABLE omraader.omraade\_typer (  
 type\_navn character varying NOT NULL,  
 passiv boolean NOT NULL DEFAULT false,  
 beskrivelse character varying,  
 CONSTRAINT omraade\_typer\_pk PRIMARY KEY (type\_navn)  
);

-- Table: arbejds\_omraader  
CREATE TABLE omraader.arbejds\_omraader (  
 omraade\_navn character varying NOT NULL,  
 omraade\_type character varying NOT NULL,  
 passiv boolean NOT NULL DEFAULT false,  
 beskrivelse character varying,  
 geom public.geometry(MultiPolygon,25832),  
 CONSTRAINT arbejds\_omraader\_pk PRIMARY KEY (omraade\_navn)  
);

ALTER TABLE omraader.arbejds\_omraader ADD CONSTRAINT arbejds\_omraader\_omraade\_typer\_fk  
 FOREIGN KEY (omraade\_type)  
 REFERENCES omraader.omraade\_typer (type\_navn);

-- Table: entreprise\_typer  
CREATE TABLE prissaetning.entreprise\_typer (  
 entreprise\_type character varying NOT NULL,  
 passiv boolean NOT NULL DEFAULT false,  
 beskrivelse character varying,  
 CONSTRAINT entreprise\_typer\_pk PRIMARY KEY (entreprise\_type)  
);

-- Table: enterpriser  
CREATE TABLE prissaetning.entrepriser (  
 id uuid NOT NULL DEFAULT gen\_random\_uuid(),  
 kontakt\_id uuid NOT NULL,  
 entreprise\_type character varying NOT NULL,  
 periode\_start date NOT NULL,  
 periode\_slut date NOT NULL,  
 CONSTRAINT entrepriser\_pk PRIMARY KEY (id)  
);

ALTER TABLE prissaetning.entrepriser ADD CONSTRAINT entrepriser\_entreprise\_typer\_fk  
 FOREIGN KEY (entreprise\_type)  
 REFERENCES prissaetning.entreprise\_typer (entreprise\_type);

ALTER TABLE prissaetning.entrepriser ADD CONSTRAINT entrepriser\_kontakter\_fk  
 FOREIGN KEY (kontakt\_id)  
 REFERENCES kontakter.kontakter (id);

-- Table: aktivitets\_typer  
CREATE TABLE aktiviteter.aktivitets\_typer (  
 gruppe\_navn character varying NOT NULL,  
 basis\_type character varying(1) NOT NULL,  
 type\_navn character varying NOT NULL,  
 passiv boolean NOT NULL DEFAULT false,  
 beskrivelse character varying,  
 CONSTRAINT aktivitets\_typer\_pk PRIMARY KEY (gruppe\_navn,type\_navn,basis\_type)  
);

-- Table: aktivitets\_priser

CREATE TABLE prissaetning.aktivitets\_priser (  
 gruppe\_navn character varying NOT NULL,  
 basis\_type character varying(1) NOT NULL,  
 type\_navn character varying NOT NULL,  
 entreprise\_id uuid NOT NULL,  
 enhedspris numeric(9,2) NOT NULL,  
 CONSTRAINT aktivitets\_priser\_pk PRIMARY KEY  
 (gruppe\_navn, type\_navn, basis\_type, entreprise\_id)

);

ALTER TABLE prissaetning.aktivitets\_priser ADD CONSTRAINT  
aktivitets\_priser\_aktivitets\_typer\_fk  
 FOREIGN KEY (gruppe\_navn, type\_navn, basis\_type)  
 REFERENCES aktiviteter.aktivitets\_typer (gruppe\_navn, type\_navn, basis\_type);

ALTER TABLE prissaetning.aktivitets\_priser ADD CONSTRAINT aktivitets\_priser\_entrepriser\_fk  
 FOREIGN KEY (entreprise\_id)  
 REFERENCES prissaetning.entrepriser (id);

ALTER TABLE aktiviteter.aktivitets\_typer ADD CONSTRAINT  
aktivitets\_typer\_elementgruppe\_typer\_fk  
 FOREIGN KEY (gruppe\_navn, basis\_type)  
 REFERENCES elementer.elementgruppe\_typer (gruppe\_navn, basis\_type);

-- Table: aktivitetsgruppe  
CREATE TABLE aktiviteter.aktivitetsgruppe (  
 id uuid NOT NULL DEFAULT gen\_random\_uuid(),  
 entreprise\_id uuid NOT NULL,  
 kontakt\_id uuid NOT NULL,  
 CONSTRAINT aktivitetsgruppe\_pk PRIMARY KEY (id)  
);

ALTER TABLE aktiviteter.aktivitetsgruppe ADD CONSTRAINT aktivitetsgruppe\_entrepriser\_fk   
 FOREIGN KEY (entreprise\_id)  
 REFERENCES prissaetning.entrepriser (id);

ALTER TABLE aktiviteter.aktivitetsgruppe ADD CONSTRAINT aktivitetsgruppe\_kontakter\_fk  
 FOREIGN KEY (kontakt\_id)  
 REFERENCES kontakter.kontakter (id);

-- Table: aktiviteter

CREATE TABLE aktiviteter.element\_aktiviteter (  
 id uuid NOT NULL DEFAULT gen\_random\_uuid(),  
 gruppe\_navn character varying NOT NULL,  
 basis\_type character varying(1) NOT NULL,  
 type\_navn character varying NOT NULL,  
 element\_flade\_id uuid,  
 element\_linje\_id uuid,  
 element\_punkt\_id uuid,  
 gruppe\_id uuid NOT NULL,  
 CONSTRAINT aktiviteter\_pk PRIMARY KEY (id)  
);

ALTER TABLE aktiviteter.element\_aktiviteter ADD CONSTRAINT element\_aktiviteter\_aktivitets\_typer\_fk  
 FOREIGN KEY (gruppe\_navn, type\_navn, basis\_type)  
 REFERENCES aktiviteter.aktivitets\_typer (gruppe\_navn, type\_navn, basis\_type);

ALTER TABLE aktiviteter.element\_aktiviteter ADD CONSTRAINT element\_aktiviteter\_aktivitets\_grupper\_fk  
 FOREIGN KEY (gruppe\_id)  
 REFERENCES aktiviteter.aktivitetsgruppe (id);

ALTER TABLE aktiviteter.element\_aktiviteter ADD CONSTRAINT element\_aktiviteter\_elementer\_flader\_fk  
 FOREIGN KEY (element\_flade\_id)  
 REFERENCES elementer.elementer\_flader (id);

ALTER TABLE aktiviteter.element\_aktiviteter ADD CONSTRAINT element\_aktiviteter\_elementer\_linjer\_fk  
 FOREIGN KEY (element\_linje\_id)  
 REFERENCES elementer.elementer\_linjer (id);

ALTER TABLE aktiviteter.element\_aktiviteter ADD CONSTRAINT element\_aktiviteter\_elementer\_punkter\_fk  
 FOREIGN KEY (element\_punkt\_id)  
 REFERENCES elementer.elementer\_punkter (id);