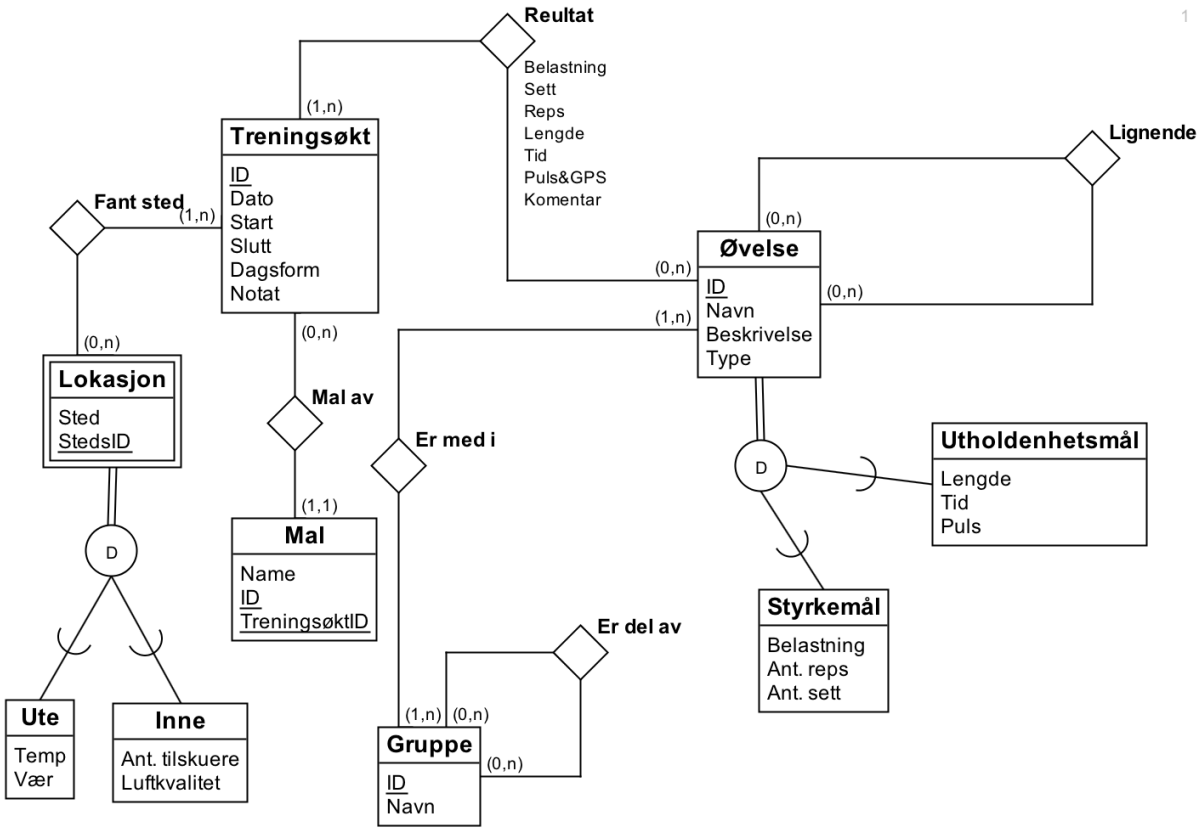
Gruppe: Andreas Strand, Jonas Jam, Truls Elgaaen, Michael Hoyer

ER-skjema:



Oppg. A del B:

Beskrivelse:

1. Registrere en treningsøkt med tilhørende data, dvs. med øvelser man har gjort, samt hvordan selve treningen har gått.
-En treningsøkt består av flere øvelser, som man har en plan for hvor stor belastning man skal ha. Dette vil si at hver øvelse er unik, med tanke på for eksempel målbelastning eller mål for antall sett for en øvelse. Det som faktisk blir gjennomført av belastning og antall sett blir lagret som et resultat, med en kommentar på hver øvelse hvordan den ble gjennomført. Treningsøkten blir evaluert i sin helhet på dagsform. Resultatet til en trengingsøkt vil være unikt, gitt de to foreign key-ene til øvelsesID og treningsøktID.

2. Lage/se oversikt over kjente treninger/øvelser, sette opp nye mål, og vite hvilke mål man har hatt.
-Med kjente øvelser mener vi øvelser man har gjort/gjennomført på et tidligere tidspunkt. Man kan hente ut en oversikt over disse ved å hente ut hvilke øvelser som er registrert blant resultatene. Det

samme kan man gjøre med mål man har tatt. Man kan sette opp nye mål ved å enten se på siste resultatet som ble registrert på en øvelse eller det beste.

3. Se progresjon for en bestemt trening/øvelse over en periode, samt hvilke mål man har hatt.
-For hver "Treningsøkt" er det en relasjon til øvelsene som ble gjort og til "Resultat" ved gitt øvelse. For hver øvelse kan man hente informasjon om resultat for en gitt dato. Dette kan fremstilles grafisk for å vise progresjon.

4. Se differensen mellom et bestemt resultat og det beste resultatet i løpet av siste uke, måned eller tre måneder, samt forskjellen mellom det og målet som har vært aktivt i den siste perioden.
-Ved å koble resultater og treningsøkter sammen kan man velge seg et resultat og sammenligne med andre resultater som kan filtreres på bl.a. tidsperioder ved å se på dato.

5. Kunne kopiere en bestemt treningsøkt over til en ny mal. Hver mal skal kunne registreres med et navn og kunne brukes til å registrere en ny treningsøkt.
-En “Mal” har en relasjon til en “Treningsøkt” slik at hvis man vil hente ut en mal for å bruke det til en ny treningsøkt, har man en direkte kobling mellom de spesifikke øvelsene, slik at man ikke trenger å legge til hver enkelt øvelse manuelt.

6. Kunne hente ut puls- eller gps-data slik at det kan vises i et eksternt program, enten som en graf og/eller med kart.
-Puls- og gps-data lagres i "Resultat" for hver øvelse pr "Treningsøkt". Dette kan fremvises i ekstern program.

7. Lese treningsnotater samlet i en logg.
-Kommentar til hver gjennomførte øvelse kan lagres i "Resultat" og dagsform samt notat for hele økten lagres i "Treningsøkt”.

8. Legge til, omorganisere og slette øvelser, grupper og delgrupper.
-Dette gjøres i SQL og kommer derfor ikke frem av ER-skjemaet.

Oppg. B del B:

En beskrivelse av hvordan SQL-scriptet støtter modellen i deloppgave 1, e.g. hvor ble det av alle relasjonsklassene og kardinalitetene?
-Hver entitetsklasse er nå blitt en tabell. En relasjon er her en tabell som linker to entitetsklasser, enten som en egen tabell (mange til mange, __) eller som et felt i entitetene (en-til-en). I tillegg kan de inneholde ekstra informasjon som er unik til relasjonen.
Kardinalitet kommer ikke veldig tydelig frem, men for eksempel å kunne ha 0 og å måtte ha 1 verdi av noe er det en klar forskjell på, nemlig om man må ha NOT NULL eller ikke på en foreign key.

SQL-script:

```
CREATE TABLE treningsokt (
        id INT,
    dato DATE,
        tidStart TIME,
    tidSlutt TIME,
    dagsform INT,
    notat VARCHAR(400),

    PRIMARY KEY(id),
    FOREIGN KEY(lokasjon) REFERENCES lokasjon(id),

    CHECK(dagsform<11 AND dagsform>0)
);

CREATE TABLE resultat (
        treningsoktID INT NOT NULL,
    ovelseID INT NOT NULL,
    belastning INT,
    reps INT,
    sett INT,
    lengde INT,
    tid INT,
    pulsOgGps TEXT,
    kommentar VARCHAR(400),

    FOREIGN KEY(treningsoktID) REFERENCES treningsokt(id),
    FOREIGN KEY(ovelseID) REFERENCES ovelse(id)
);

CREATE TABLE ovelse (
        id INT NOT NULL,
    navn VARCHAR(50),
    beskrivelse VARCHAR(500),

    PRIMARY KEY(id)
);

CREATE TABLE lignendeOvelse (
        ovelseID INT NOT NULL,
    lignendeOvelseID INT NOT NULL,

    FOREIGN KEY(ovelseID) REFERENCES ovelse(id),
    FOREIGN KEY(ligendeOvelseID) REFERENCES ovelse(id)
);

CREATE TABLE styrkemaal (
        ovelseID INT NOT NULL,
    belastning INT,
    reps INT,
    sett INT,
```

```
    FOREIGN KEY(ovelseID) REFERENCES ovelse(id)
);

CREATE TABLE utholdenshetsmaal (
        ovelseID INT NOT NULL,
    lengde INT,
    tid INT,
    puls INT,

    FOREIGN KEY(ovelseID) REFERENCES ovelse(id)
);

CREATE TABLE gruppe (
        id INT NOT NULL,
    navn VARCHAR(40),

    PRIMARY KEY(id)
);

CREATE TABLE gruppeDelAv (
        gruppeID INT NOT NULL,
    overGruppeID INT NOT NULL,

    FOREIGN KEY(gruppeID) REFERENCES gruppe(id),
    FOREIGN KEY(overGruppeID) REFERENCES gruppe(id)
        );

CREATE TABLE ovelseOgGruppe (
        gruppeID INT NOT NULL,
    ovelseID INT NOT NULL,

    FOREIGN KEY(gruppeID) REFERENCES gruppe(id),
    FOREIGN KEY(ovelseID) REFERENCES ovelse(id)
);

CREATE TABLE lokasjon (
        id INT NOT NULL,
    sted VARCHAR(40),

    PRIMARY KEY(id)
);

CREATE TABLE inne (
        lokasjonID INT NOT NULL,
    tilskuere INT,
    luftkvalitet INT,

    FOREIGN KEY(lokasjonID) REFERENCES lokasjon(id)
);

CREATE TABLE ute (
        lokasjonID INT NOT NULL,
```

```
    temp INT,
    vaer VARCHAR(100),

    FOREIGN KEY(lokasjonID) REFERENCES lokasjon(id)
);

CREATE TABLE treningsLokasjon (
        treningsoktID INT NOT NULL,
    lokasjonID INT NOT NULL,

    FOREIGN KEY(treningsoktID) REFERENCES treningsokt(id),
    FOREIGN KEY(lokasjonID) REFERENCES lokasjon(id)
);

CREATE TABLE mal (
        id INT NOT NULL,
    treningsoktID INT NOT NULL,
    navn VARCHAR(50),

    PRIMARY KEY(id),
    FOREIGN KEY(treningsoktID) REFERENCES treningsokt(id)
);
```