SQL-Script

```
-- The following statements creates the database and tables for the Alarmsystem Database

CREATE DATABASE Alarmsystem
go
USE Alarmsystem
Go

CREATE TABLE SUBSCRIBER
(
Email char (30),
FirstName char(18) NOT NULL,
LastName char(18) NOT NULL,
Telephone int,
CONSTRAINT PK_SUBSCRIBER PRIMARY KEY (Email)
)

CREATE TABLE TEMPERATURE
(
[Time] datetime,
Temperature float NOT NULL,
CONSTRAINT PK_TEMPERATURE PRIMARY KEY ([Time])
)

CREATE TABLE ALARMTYPE
(
Alarm char(30),
Limit float,
[Description] varchar(100),
CONSTRAINT PK_ALARMTYPE PRIMARY KEY (Alarm)
```

```
)

CREATE TABLE ALARM
(
Alarm char(30),
[Time] datetime,
CONSTRAINT PK_ALARM PRIMARY KEY (Alarm, [Time]),
CONSTRAINT FK_ALARMTYPE_ALARM FOREIGN KEY (Alarm)
REFERENCES ALARMTYPE (Alarm)
)

CREATE TABLE SUBSCRIBESTO
(
Alarm char(30),
Email char(30),
CONSTRAINT PK_SUBSCRIBESTO PRIMARY KEY (Alarm, Email),
CONSTRAINT FK_ALARMTYPE_SUBSCRIBESTO FOREIGN KEY (Alarm)
REFERENCES ALARMTYPE (Alarm),
CONSTRAINT FK_SUBSCRIBER_SUBSCRIBESTO FOREIGN KEY (Email)
REFERENCES SUBSCRIBER (Email)
)

/* Metode for å legge til en ny abbonent */
CREATE PROCEDURE NewSubscriber
@Email char(30), @FirstName char(18), @LastName char(18), @Telephone int
AS
INSERT INTO SUBSCRIBER
VALUES (@Email, @FirstName, @LastName, @Telephone)
GO

/* Metode for å lagre en ny temperaturmåling */
```

```
CREATE PROCEDURE MeasureTemp
@Temperature float
AS
INSERT INTO TEMPERATURE
VALUES (CURRENT_TIMESTAMP ,@Temperature)
GO

/* Metode for å lagrer en ny alarm i alarmhistorikken */
CREATE PROCEDURE NewAlarm
@Alarm char(30)
AS
INSERT INTO ALARM
VALUES (@Alarm, CURRENT_TIMESTAMP)
GO

/* Metode for å oppdatere grenseverdiene for alarmene */
CREATE PROCEDURE UpdateLimit
@AlarmType char(30), @NewLimit float
AS
UPDATE ALARMTYPE
SET Limit = @NewLimit
WHERE ALARMTYPE.Alarm = @AlarmType
GO

/* Metode som returnerer alarmgrense for en gitt alarm */
CREATE PROCEDURE GetLimit
@AlarmType char(30)
AS
SELECT Limit
FROM ALARMTYPE
WHERE Alarm = @AlarmType
```

```
GO

/* Metode som returnerer beskrivelsen til en gitt alarm */
CREATE PROCEDURE GetDescription
@AlarmType char(30)
AS
SELECT [Description]
FROM ALARMTYPE
WHERE Alarm = @AlarmType
GO

-- Metode som oppdaterer en eksisterende abonnent med nye verdier
CREATE PROCEDURE UpdateSubscriber
@Email char(30), @NewEmail char(30), @FirstName char(18), @LastName char(18), @Telephone int
AS
ALTER TABLE SUBSCRIBESTO NOCHECK CONSTRAINT ALL --skrur av FK før oppdatering
UPDATE SUBSCRIBER
SET Email = @NewEmail,
FirstName = @FirstName,
LastName = @LastName,
Telephone = @Telephone
WHERE Email = @Email
UPDATE SUBSCRIBESTO
SET Email = @NewEmail
WHERE Email = @Email
ALTER TABLE SUBSCRIBESTO CHECK CONSTRAINT ALL --setter FK tilbake etter oppdatering
GO

drop procedure UpdateSubscriber

-- Metode som sletter en abonent
```

```
CREATE PROCEDURE DeleteSubscriber
@Email char(30)
AS
DELETE SUBSCRIBESTO WHERE Email = @Email
DELETE SUBSCRIBER  WHERE Email = @Email
GO

-- Metode som legger en abonent i abonentliste
CREATE PROCEDURE NewSubscribesTo
@Email char(30), @Alarmtype char(30)
AS
INSERT INTO SUBSCRIBESTO
VALUES (@Alarmtype, @Email)
GO

CREATE PROCEDURE UnsubscribesTo
@Email char(30), @Alarmtype char(30)
AS
DELETE SUBSCRIBESTO
WHERE Email = @Email
AND ALARM = @Alarmtype
GO

-- Inserts the types of alarms needed
INSERT INTO ALARMTYPE
VALUES ('HighTemp', 30.0, 'Måleverdi for temperatur er over øvre grense'),
('LowTemp', 15.0, 'Måleverdi for temperature er under nedre grense'),
('LowBattery', 20, 'Batterispenningen er under nedre grense'),
('NoCharge', null, 'Datamaskinens strømforsyning er ikke tilkoblet'),
('ComFault', null, 'Kommunikasjon med Arduino feilet'),
('Motion', null, 'Motion detected!')
```