-- (Original script templates by Kari Silpiö 2012)

-- Juhani Välimäki and Markku Kuitunen.

-- Fisherman database creation scripts 2014-02-07

-- Keywords that shuold be avoided: user, table, name, text

-- In other systems even: description

-- Updated by Mirya Nezvitskaya

CREATE PROCEDURE [Create\_Tables]

AS

/\*

EXEC sp00\_DropTables;

\*/

-- Juhani simplified, all data VARCHAR(100) or (MAX), all integers INTEGER

-- To be decided: Removing NOT NULL from some fields of some tables

CREATE TABLE Fisherman

(

fId INTEGER NOT NULL, -- NOT NULL IDENTITY

firstName NVARCHAR(100) NOT NULL,

lastName NVARCHAR(100) NOT NULL,

phone NVARCHAR(100) NOT NULL,

eMail NVARCHAR(100) NOT NULL,

gender CHAR(1) NOT NULL,

passwordHashFields NVARCHAR(100) NOT NULL,

CONSTRAINT pk\_Fisherman PRIMARY KEY (fId),

CONSTRAINT chk\_f\_gender

CHECK (gender IN ('F','M')),

CONSTRAINT unique\_Fisherman\_eMail UNIQUE (eMail)

);

CREATE TABLE LureType

(

ltId INTEGER NOT NULL, -- NOT NULL IDENTITY

typename NVARCHAR(100) NOT NULL,

description NVARCHAR(200) NOT NULL,

CONSTRAINT pk\_LureType PRIMARY KEY (ltId),

CONSTRAINT chk\_LureType\_typename

CHECK (typename NOT IN (''))

);

CREATE TABLE FishSpecies

(

fiId INTEGER NOT NULL, -- NOT NULL IDENTITY

fishname NVARCHAR(100) NOT NULL,

description NVARCHAR(200) NOT NULL,

CONSTRAINT pk\_FishSpecies PRIMARY KEY (fiId),

CONSTRAINT chk\_FishSpecies\_fishname

CHECK (fishname NOT IN (''))

);

CREATE TABLE FishingMethod

(

fmId INTEGER NOT NULL, -- NOT NULL IDENTITY

methodname NVARCHAR(100) NOT NULL,

description NVARCHAR(200) NOT NULL,

CONSTRAINT pk\_FishingMethod PRIMARY KEY (fmId),

CONSTRAINT chk\_FishingMethod\_methodname

CHECK (methodname NOT IN (''))

);

CREATE TABLE Water

(

wId INTEGER NOT NULL, -- NOT NULL IDENTITY

name NVARCHAR(100) NOT NULL,

municipality NVARCHAR(100) NOT NULL,

description NVARCHAR(200) NOT NULL,

CONSTRAINT pk\_Water PRIMARY KEY (wId),

CONSTRAINT chk\_Water\_name

CHECK (name NOT IN ('')),

CONSTRAINT chk\_Water\_municipality

CHECK (municipality NOT IN (''))

);

CREATE TABLE GeoCoordinates

(

gcId INTEGER NOT NULL, -- NOT NULL IDENTITY

latitude DECIMAL(10,7) NOT NULL,

longitude DECIMAL(10,7) NOT NULL,

CONSTRAINT pk\_GeoCoordinates PRIMARY KEY (gcId)

);

CREATE TABLE LocationMarking

(

lmId INTEGER NOT NULL, -- NOT NULL IDENTITY

sublocation NVARCHAR(100) NOT NULL,

compaslocation NVARCHAR(100) NOT NULL,

description NVARCHAR(200) NOT NULL,

wId INTEGER NOT NULL,

gcId INTEGER,

CONSTRAINT pk\_LocationMarking PRIMARY KEY (lmId),

CONSTRAINT chk\_LocationMarking\_sublocation

CHECK (sublocation NOT IN ('')),

CONSTRAINT fk\_LocationMarking\_Water FOREIGN KEY (wId)

REFERENCES Water(wId),

CONSTRAINT fk\_LocationMarking\_GeoCoordinates FOREIGN KEY (gcId)

REFERENCES GeoCoordinates(gcId)

);

CREATE TABLE Lure

(

lId INTEGER NOT NULL, -- NOT NULL IDENTITY

name NVARCHAR(100) NOT NULL,

description NVARCHAR(200) NOT NULL,

ltId INTEGER NOT NULL,

CONSTRAINT pk\_Lure PRIMARY KEY (lId),

CONSTRAINT fk\_Lure\_LureType FOREIGN KEY (ltId)

REFERENCES LureType(ltId)

);

CREATE TABLE Catch

(

cId INTEGER NOT NULL, -- NOT NULL IDENTITY

weight DECIMAL(10,7) NOT NULL,

lureweight INTEGER NOT NULL,

depth INTEGER NOT NULL,

description NVARCHAR(200) NOT NULL,

lId INTEGER NOT NULL,

fiId INTEGER NOT NULL,

CONSTRAINT pk\_Catch PRIMARY KEY (cId),

CONSTRAINT fk\_Catch\_Lure FOREIGN KEY (lId)

REFERENCES Lure(lId),

CONSTRAINT fk\_Catch\_FishSpecies FOREIGN KEY (fiId)

REFERENCES FishSpecies(fiId)

);

CREATE TABLE FishingSession

(

fsId INTEGER NOT NULL, -- NOT NULL IDENTITY

date DATE NOT NULL,

description NVARCHAR(200) NOT NULL,

fId INTEGER NOT NULL,

fmId INTEGER NOT NULL,

cId INTEGER,

lmId INTEGER NOT NULL,

CONSTRAINT pk\_FishingSession PRIMARY KEY (fsId),

CONSTRAINT fk\_FishingSession\_Fisherman FOREIGN KEY (fId)

REFERENCES Fisherman(fId),

CONSTRAINT fk\_FishingSession\_FishingMethod FOREIGN KEY (fmId)

REFERENCES FishingMethod(fmId),

CONSTRAINT fk\_FishingSession\_Catch FOREIGN KEY (cId)

REFERENCES Catch(cId),

CONSTRAINT fk\_FishingSession\_LocationMarking FOREIGN KEY (lmId)

REFERENCES LocationMarking(lmId)

);

-- Finally, display a message

IF (@@Error = 0)

BEGIN

PRINT '============================='

PRINT ' TABLES CREATED SUCCESSFULLY.'

PRINT '============================='

PRINT ' '

END

RETURN 0

-- (Original script templates by Kari Silpiö 2012)

-- Juhani Välimäki and Markku Kuitunen.

-- Fisherman database creation scripts 2014-02-07

CREATE PROCEDURE [Drop\_Tables]

AS

/\* Juhani added a template for creating more drops

IF OBJECT\_ID('') IS NOT NULL

BEGIN

DROP TABLE xxxxxx;

END

\*/

-- \*\*\* Trace log tables dropped at the beginning!!!

IF OBJECT\_ID('FishingSession') IS NOT NULL

BEGIN

DROP TABLE FishingSession;

END

IF OBJECT\_ID('Catch') IS NOT NULL

BEGIN

DROP TABLE Catch;

END

IF OBJECT\_ID('Lure') IS NOT NULL

BEGIN

DROP TABLE Lure;

END

IF OBJECT\_ID('LocationMarking') IS NOT NULL

BEGIN

DROP TABLE LocationMarking;

END

IF OBJECT\_ID('GeoCoordinates') IS NOT NULL

BEGIN

DROP TABLE GeoCoordinates;

END

IF OBJECT\_ID('Water') IS NOT NULL

BEGIN

DROP TABLE Water;

END

IF OBJECT\_ID('FishingMethod') IS NOT NULL

BEGIN

DROP TABLE FishingMethod;

END

IF OBJECT\_ID('FishSpecies') IS NOT NULL

BEGIN

DROP TABLE FishSpecies;

END

IF OBJECT\_ID('LureType') IS NOT NULL

BEGIN

DROP TABLE LureType;

END

IF OBJECT\_ID('Fisherman') IS NOT NULL

BEGIN

DROP TABLE Fisherman;

END

-- Finally, display a message

IF (@@Error = 0)

BEGIN

PRINT '================================='

PRINT ' ALL TABLES DROPPED SUCCESSFULLY.'

PRINT '================================='

PRINT ' '

END

ELSE

BEGIN

PRINT '=============================='

PRINT ' See the error messages above!'

PRINT '=============================='

PRINT ' '

END

RETURN 0

-- (Original script templates by Kari Silpiö 2012)

-- Juhani Välimäki and Markku Kuitunen.

-- Fisherman database creation scripts 2014-02-07

-- Updated by Mirya Nezvitskaya

CREATE PROCEDURE [Insert\_Test\_Data]

AS

INSERT INTO Fisherman

(fId, firstName, lastName, phone,

eMail, gender, passwordHashFields)

VALUES

(1, 'Frank', 'Zappa', '09-987654321',

'fz@email.com', 'M', '2A5D0D7A231415'),

(2, 'Vladimir', 'Putin', '09-123456789',

'vp@email.com', 'M', 'FDGHFK4FBVGFKG'),

(3, 'Angela', 'Merkel', '09-354546567',

'am@email.com', 'F', 'DFKJDR435456GF'),

(4, 'Bill', 'Gates', '09-457845784',

'bg@email.com', 'M', 'DFKDKDFJGKFER4'),

(5, 'George', 'Bush', '09-34345454',

'gbm@email.com', 'M', 'DFKFGHFJK4TDFG'),

(6, 'Ronnie James', 'Dio', '09-4546654',

'rgd@email.com', 'M', 'RGTR5JHGRJRGGH ');

INSERT INTO LureType

(ltId, typename, description )

VALUES

(1, 'Jigs', ' A jig can catch about every game fish there is, and are inexpensive to boot.' ),

(2, 'Jerks', 'Great for fly fishing.' ),

(3, 'Spoons', 'Spoons are curved metal lures. The first spoons were just that, spoons with the handle broken off.' ),

(4, 'Spinners', 'Spinners are a great beginner lure because they are so easy to use. They are essentially a metal shaft with spinning blade.' ),

(5, 'Soft Plastic Baits', 'Soft plastic baits encompass a variety of different lures, mostly used for bass fishing. The classic soft plastic bait is the worm.' ),

(6, 'Plugs', 'Plugs are constructed from hollow plastic or wood to resemble baitfish, frogs or other prey. They usually sport two or three treble hooks..' );

INSERT INTO FishSpecies

(fiId , fishname, description )

VALUES

(1, 'Salmon', 'Common name for fish of the group Salmonidae. It is pink.'),

(2, 'Pike', 'North fish.'),

(3, 'Trout', 'Common name for fish of the group Salmonidae. Colour is gray-golden.'),

(4, 'Toutain', 'This delicious fish can be found in Lake Inarijärvi and Lake Kilpisjärvi, as well as in the mountain lakes in the most northerly parts of Lapland.'),

(5, 'Silakka', 'A small fish, measuring between 14 and 18 cm, Baltic Herring is the most plentiful fish to be found in the coastal waters of Finland, and is very much a staple food in the Finnish diet.'),

(6, 'Lahna', 'One of Finlands largest cyprinids, the most viable stocks are in the lakes of southern and central Finland, and in the shallow sea bays.');

INSERT INTO FishingMethod

(fmId, methodname , description)

VALUES

(1, 'Casting', ' Casting means throwing a lure that imitates a fish, some 15-30 meters away and then reeling it back in so that it swims like a fish'),

(2, 'Fly Fishing', 'You typically do this standing in the water. Typical is the back, forward, back, forward movement of the visible fishing line before finally releasing the throw.'),

(3, 'Net Fishing', ' You put a long, usually straight net with weights (verkon paino) at the bottom and floaters at the top (verkon koho).'),

(4, 'Winter Fishing', 'You take a big ice drill, walk to the lake or sea ice, drill a hole through the ice and sit down to fish by moving the lure up and down. There are two kinds of lures'),

(5, 'Trolling', 'You use the same equipment as in Casting, but you fix the fishing rod at the end of the boat and either row or use small motor to move the boat so that the lure "swims" following the boat. ');

INSERT INTO Water

(wId , name, municipality, description )

VALUES

(1, 'Saimaa' , 'Joensuu', 'Beautiful lake for pikes'),

(2, 'Nuuksio', 'Espoo', 'Biggest lake in Uusimaa, great for small fishes.'),

(3, 'Lappi Lake', 'Lapland', 'Great for winter fishing and net fishing'),

(4, 'Aapajärvi' , 'Lapland', 'Great for winter fishing. '),

(5, 'Elämäjärvi' , 'Keski-Suomi', 'Small lake for silakka fishing'),

(6, 'Jääsjärvi' , 'Joutsa', 'Beautiful lake for fishing.');

INSERT INTO GeoCoordinates

(gcId, latitude, longitude)

VALUES

(1, 61.25, 28.25),

(2, 60.45, 27.56),

(3, 26.78, 45.67),

(4, 23.78, 45.67),

(5, 54.34, 56.49),

(6, 23.88, 12.89);

INSERT INTO LocationMarking

(lmId, sublocation , compaslocation, description ,

wId , gcId)

VALUES

(1, 'Joensuu', '-', 'Lots of pikes.', 2 , null

),

(2, 'Espoo', '2W5', 'Fishing for pikes.',

3, 2),

(3, 'Lappi', '4W5', 'For winter fishing',

4, null),

(4, 'Savonlinna', '6W5', 'Great for trolling and casting' ,

6, 4);

INSERT INTO Lure

(lId , name, description , ltId)

VALUES

(1, 'blue jig', 'Good for catching pike ', 1),

(2, 'white jig', '-', 1),

(3, 'colourful jerk', 'Baught in S-Market for 2 euros', 2),

(4, 'red small spinner', 'Used for lohi. ', 4),

(5, 'red plug', 'Great plug ', 6),

(6, 'colourful plug', 'Baught 2 euros ago for fly fishing ', 6);

INSERT INTO Catch

(cId , weight , lureweight, depth,

description , lId , fiId)

VALUES

(1, 7.8, 12, 1,

'Quite eassy to catch, caught in first 5 minutes. ', 1, 1),

(2, 2.8, 5, 2,

'- ', 1, 2),

(3, 4.18, 10, 1,

'Lure really helped', 2, 3),

(4, 3.45, 3, 2,

'Was difficult to catch, now gonna make some soup.', 3, 5),

(5, 12.1, 6, 3,

'-', 4, 5),

(6, 2.23, 7, 10,

'Great day for fishing!', 4, 6);

INSERT INTO FishingSession

(fsId, date, description, fId ,

fmId, cId , lmId)

VALUES

(1, '2013-12-03', 'Went fishing with friends, caught nothing.', 1,

1, null, 1),

(2, '2013-05-05', 'Was raining, baught several pikes.', 2,

2, 2, 4),

(3, '2013-07-04', 'Caught 2 kilo of fish!', 3,

4, 2, 3),

(4, '2014-12-03', 'Went fishing with friends, caught nothing.', 1,

1, null, 3),

(5, '2013-01-10', 'Went fishing with friends, caught lots of fish.', 1,

1, 2, 1);

-- Finally, display a message (not robust error handling...)

IF (@@Error = 0)

BEGIN

PRINT '================================='

PRINT ' TEST DATA INSERTED SUCCESSFULLY.'

PRINT '================================='

PRINT ' '

END

ELSE

BEGIN

PRINT '================================='

PRINT ' TEST DATA INSERTION FAILED!'

PRINT ' See the error messages above.'

PRINT ' '

PRINT ' (Maybe tables do not exist yet?)'

PRINT '================================='

PRINT ' '

END

-- End –

-- (Original script templates by Kari Silpiö 2012)

-- Markku Kuitunen and Juhani Välimäki.

-- Fisherman database creation scripts 2013-03-25

CREATE PROCEDURE [Rebuild]

AS

BEGIN

SET NOCOUNT ON

EXEC Drop\_Tables

EXEC Create\_Tables

EXEC Insert\_Test\_Data

END