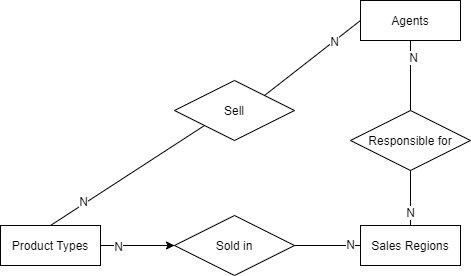
# Excercise 1

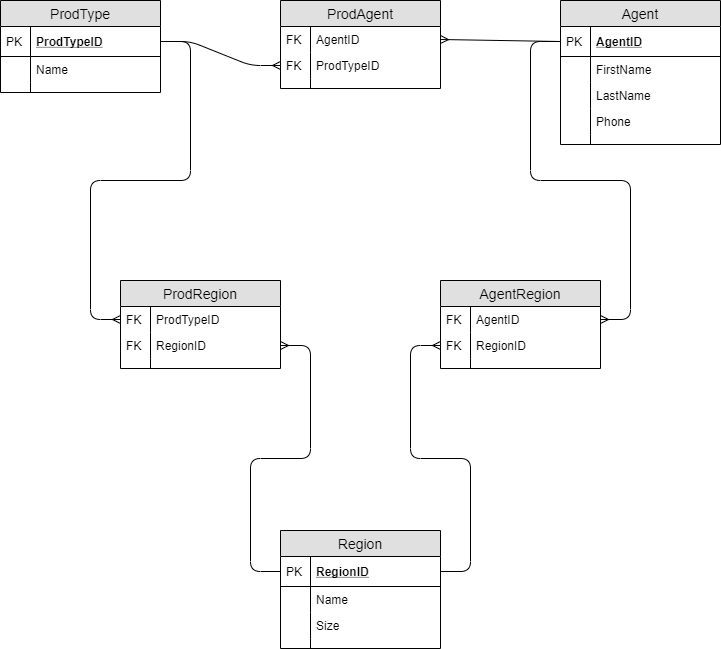
## You should create database, that contains information about sales agents, sales regions and product types. Each agent is responsible for sales in one or more regions, each region has one or more agents. Each agent is responsible for sales of one or more product types. Each product type is sold by one or more agents. Each product type is sold in one or more regions; one or more product types are sold in each region. If agent A is responsible for region R, product type P is sold in region R and agent A sells product type P, then Agent A sells product type P in region R.

## Create database schema according to these criteria. (Crate conceptual model, logical model, physical model, FD graphs for each table. You should describe normalform for each table. Tables must be in at least BCNF or you must justify why it is not so.

## Konceptual model

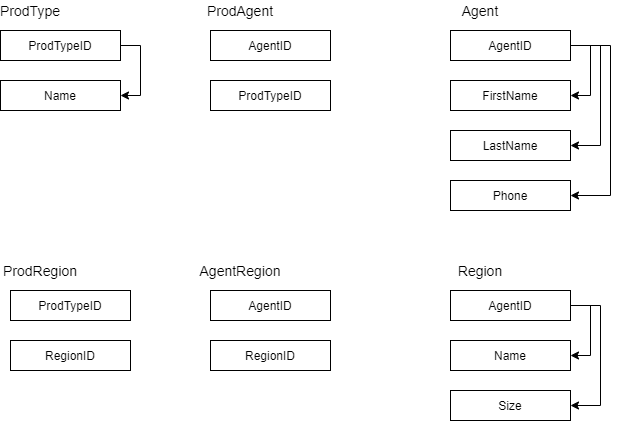


## Loģiskais modelis



All fields of relations ProdAgent, ProdRegion and AgentRegion are foreign keys and also form primary keys.

## FD Graphs



## Normalforms

All tables are in Normalform 1 because all fields are atomic.

All tables has primary keys (Primary keys for tables ProdAgent, ProdRegion and AgentRegion are combinations of both fields of theese tables); therefore all tables are in normalform 2.

All functional dependencies begins from full primary or potential key; therefore all tables are in BCNF.

## Tables create scripts

CREATE DATABASE Products

GO

USE Products

GO

CREATE TABLE Agent(

AgentID int NOT NULL,

FirstName nchar(20) NULL,

LastName nchar(20) NULL,

Phone nchar(20) NULL,

PRIMARY KEY (AgentID)

)

GO

CREATE TABLE AgentRegion(

RegionID int NULL,

AgentID int NULL

)

GO

CREATE TABLE ProdAgent(

ProdTypeID int NULL,

AgentID int NULL

)

GO

CREATE TABLE ProdRegion(

ProdTypeID int NULL,

RegionID nt NULL

)

GO

CREATE TABLE ProdType(

ProdTypeID int NOT NULL,

Name nchar(20) NULL,

PRIMARY KEY (ProdTypeID)

)

GO

CREATE TABLE Region(

RegionID int NOT NULL,

Name nchar(20) NULL,

Size int NULL,

PRIMARY KEY (RegionID)

)

GO

ALTER TABLE AgentRegion WITH CHECK ADD FOREIGN KEY(AgentID)

REFERENCES Agent(AgentID)

GO

ALTER TABLE AgentRegion WITH CHECK ADD FOREIGN KEY(RegionID)

REFERENCES Region(RegionID)

GO

ALTER TABLE ProdAgent WITH CHECK ADD FOREIGN KEY(AgentID)

REFERENCES Agent(AgentID)

GO

ALTER TABLE ProdAgent WITH CHECK ADD FOREIGN KEY(ProdTypeID)

REFERENCES ProdType(ProdTypeID)

GO

ALTER TABLE ProdRegion WITH CHECK ADD FOREIGN KEY(ProdTypeID)

REFERENCES ProdType(ProdTypeID)

GO

ALTER TABLE ProdRegion WITH CHECK ADD FOREIGN KEY(RegionID)

REFERENCES Region(RegionID)

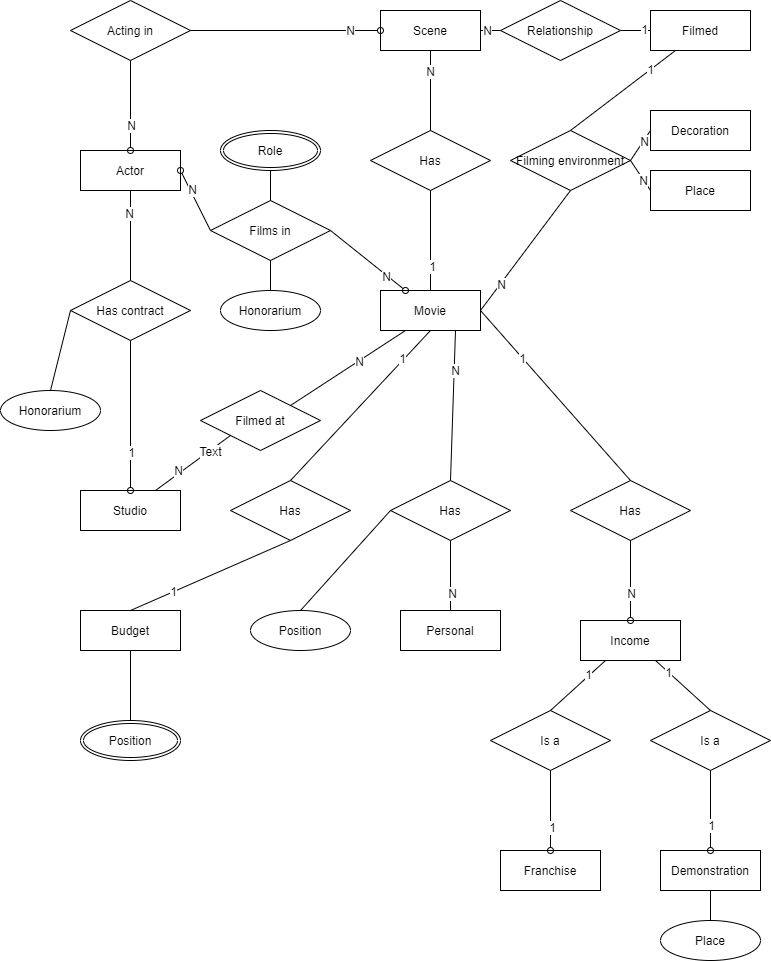
GO

# Excercise 2

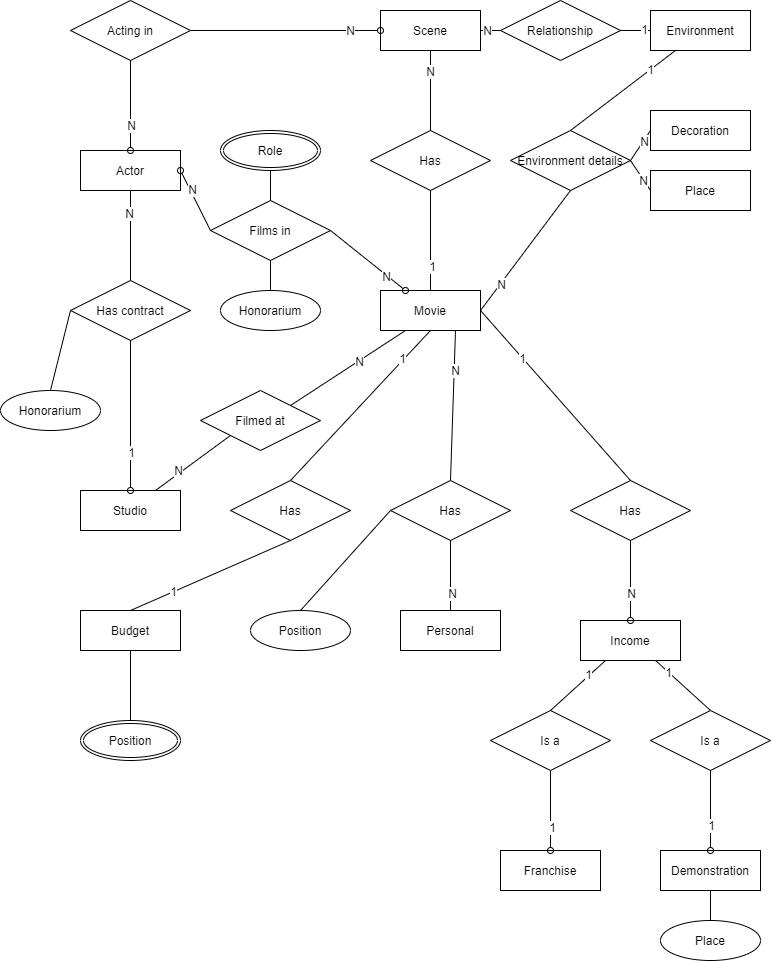
Movies are shot at studios. Studio has name and location. Actor films in movie in some role (actor can have many roles in one movie). Actor has contract with studio about filming. Actor has his main amplua in which he usually films. Actor has amount of his standard honorarium. Honorarium for each movie can vary. Movie has genre and length. Movie is shot in different places and with different decorations. Movie has director and other personal involved in movie production. Movie has budget divided by positions. Move has income from demonstration in cinemas and online film services (Netflix, Amazon etc.). Movie has income from selling souvenirs, t-shirts etc. with movie logo. Movie is divided by scenes. Each scene has place, where it is filmed. Scene has properties necessary for particular scene. Each property has name, price, description and additional attributes. Scene has actors acting in this scene.

Create database schema according to these requirements and criteria. (Crate conceptual model, logical model, physical model, FD graphs for each table. You should describe normalform for each table. Tables must be in at least BCNF or you must justify why it is not so.

## Conceptual model

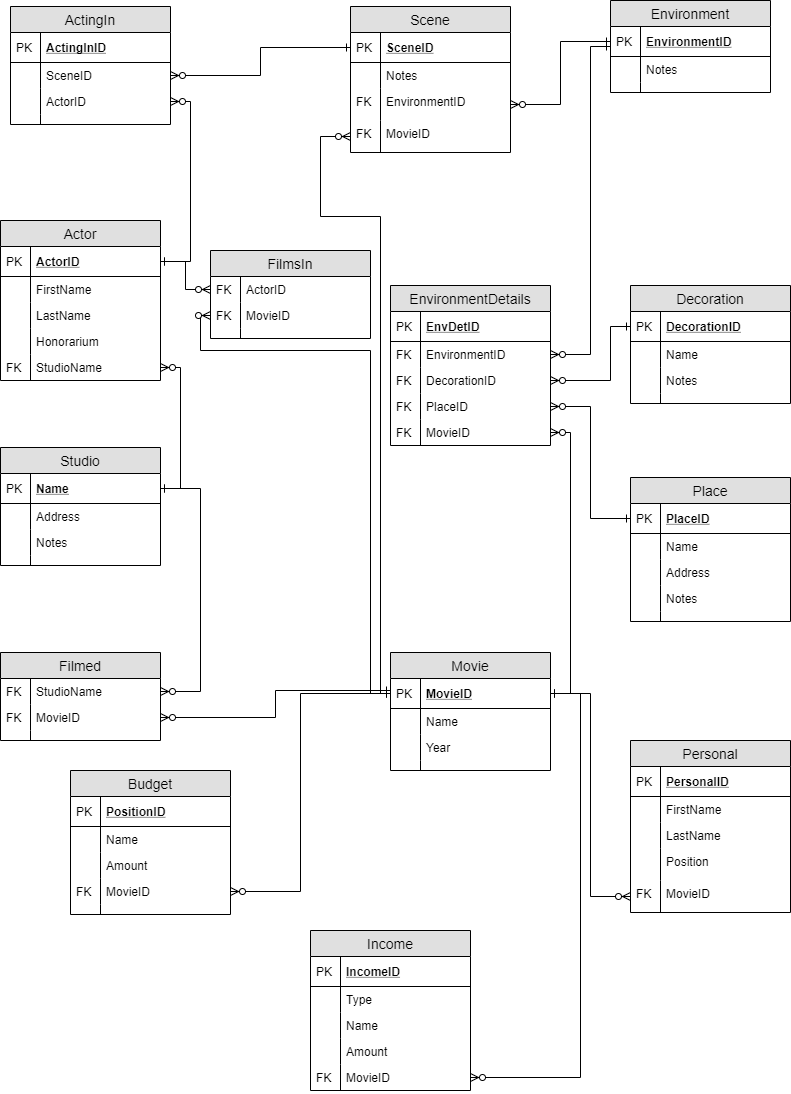


Level 0



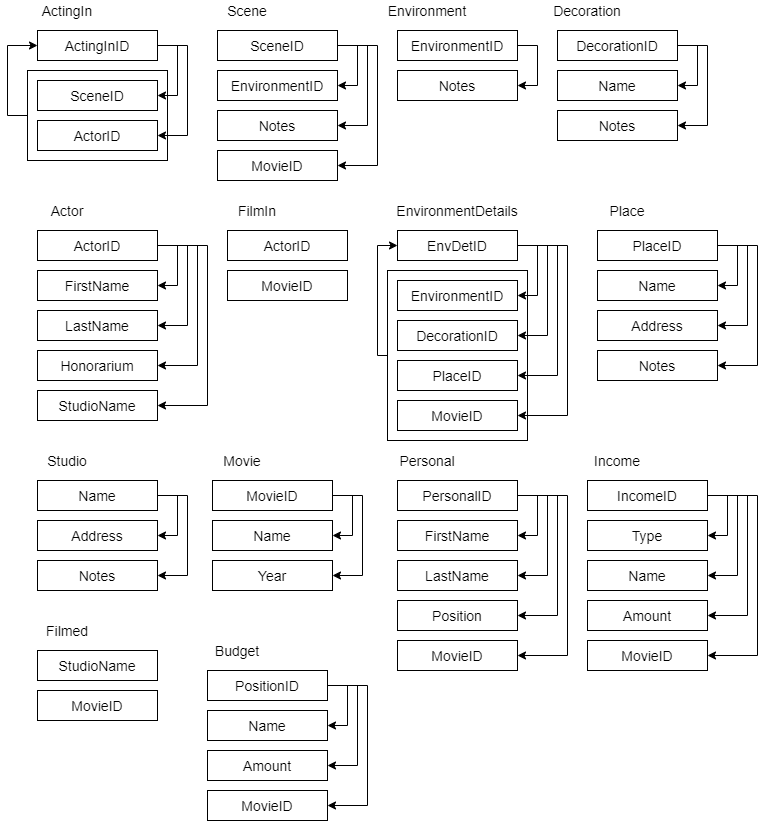
Level 1

## Logical model



All fields of relations FilmsIn and Filmed are foreign keys and also form primary keys.

## FD graphs



## Normalforms

All tables are in Normalform 1 because all fields are atomic.

All tables has primary keys (Primary keys for tables FilmIn and Filmed are combinations of both fields of theese tables); therefore all tables are in normalform 2.

All functional dependencies begins from full primary or potential key; therefore all tables are in BCNF.

## Table create scripts.

CREATE DATABASE Movies

USE Movies

CREATE TABLE Movie(

MovieID INT PRIMARY KEY,

Name NVARCHAR(100),

YEAR CHAR(4))

CREATE TABLE Personal(

PersonalID INT PRIMARY KEY,

FirstName NCHAR(20),

LastName NCHAR(20),

Position NVARCHAR(50),

MovieID INT FOREIGN KEY REFERENCES Movie(MovieID))

CREATE TABLE Income(

IncomeID INT PRIMARY KEY,

Type NCHAR(5),

Name NCHAR(20),

Amount MONEY,

MovieID INT FOREIGN KEY REFERENCES Movie(MovieID))

CREATE TABLE Budget(

PositionID INT PRIMARY KEY,

Name NCHAR(20),

Amount MONEY,

MovieID INT FOREIGN KEY REFERENCES Movie(MovieID))

CREATE TABLE Studio(

Name NVARCHAR(100) PRIMARY KEY,

Address NVARCHAR(200),

Notes NVARCHAR(2000))

CREATE TABLE Filmed(

StudioName NVARCHAR(100) FOREIGN KEY REFERENCES Studio(Name),

MovieID INT FOREIGN KEY REFERENCES Movie(MovieID))

CREATE TABLE Actor(

ActorID INT PRIMARY KEY,

FirstName NCHAR(20),

LastName NCHAR(20),

Honorarium MONEY,

StudioName NVARCHAR(100) FOREIGN KEY REFERENCES Studio(Name))

CREATE TABLE FilmsIn(

ActorID INT FOREIGN KEY REFERENCES Actor(ActorID),

MovieID INT FOREIGN KEY REFERENCES Movie(MovieID))

CREATE TABLE Place(

PlaceID INT PRIMARY KEY,

Name NVARCHAR(100),

Address NVARCHAR(200),

Notes NVARCHAR(2000))

CREATE TABLE Decoration(

DecorationID INT PRIMARY KEY,

Name NVARCHAR(100),

Notes NVARCHAR(2000))

CREATE TABLE Environment(

EnvironmentID INT PRIMARY KEY,

Notes NVARCHAR(2000))

CREATE TABLE Scene(

SceneID INT PRIMARY KEY,

Notes NVARCHAR(2000),

EnvironmentID INT FOREIGN KEY REFERENCES Environment(EnvironmentID),

MovieID INT FOREIGN KEY REFERENCES Movie(MovieID))

CREATE TABLE EnvironmentDetails(

EnvDetID INT PRIMARY KEY,

EnvironmentID INT FOREIGN KEY REFERENCES Environment(EnvironmentID),

DecorationID INT FOREIGN KEY REFERENCES Decoration(DecorationID),

PlaceID INT FOREIGN KEY REFERENCES Place(PlaceID),

MovieID INT FOREIGN KEY REFERENCES Movie(MovieID))