Relační model

# Konceptuální model

Diagram

Description automatically generated

**Person** (nationalId, phone, birth, firstName, lastName, city, street, postalCode)

**Employee** (nationalId, contractNumber, workPosition, salary)

FK: (naitonalNumber) Person(nationalId)

**Supervise** (supervised, supervisor)

FK: (supervised) Employee (nationalId)

FK: (supervisor) Employee (nationalId)

**Invoice** (invoiceNumber, employee, dateOfIssue, cost)

FK: employee Employee(nationalId)

**Customer** (nationalId)

FK: (naitonalNumber) Person(nationalId)

**Membership** (type, customer, invoice, expireDate, cost)

FK: (customer) Customer(nationalId)

FK: (invoice) Invoice(invoiceNumber)

**Trainer** (nationalId)

FK: (naitonalNumber) Person(nationalId)

**TrainersLicense** (trainer, trainersLicense)

FK: (trainer) Trainer (nationalId)

**WorkoutClass** (name, day, time, teacher, capacity)

FK: (teacher) Trainer (nationalId)

**TakeClass** (customer, workoutClass, price)

FK: (customer) Customer (nationalId)

FK: (workoutClass) WorkoutClass (name, day, time, teacher)

## SQL dotazy pro vytvoření databáze

--------- Create table ----------

CREATE TABLE IF NOT EXISTS Person (

nationalId CHAR(11) PRIMARY key,

phone CHAR(11) UNIQUE,

birth DATE CHECK (birth < now()),

firstName VARCHAR(120) NOT NULL,

lastName VARCHAR(120) NOT NULL,

city VARCHAR(120) NOT NULL,

street VARCHAR(120) NOT NULL,

postalCode char(6) NOT NULL,

CONSTRAINT nationalId\_check CHECK (nationalId ~ '^[0-9]{6}\/[0-9]{4}$'),

CONSTRAINT postalCode\_check CHECK (postalcode ~ '^[0-9]{3}[[:blank:]][0-9]{2}$'),

CONSTRAINT phone CHECK (phone ~ '^[0-9]{3}[[:blank:]][0-9]{3}[[:blank:]][0-9]{3}')

);

CREATE TABLE IF NOT EXISTS Employee (

nationalId CHAR(11) PRIMARY KEY,

contractNumber INT UNIQUE,

workPosition VARCHAR(120),

salary DECIMAL(10, 2) NOT NULL,

FOREIGN KEY (nationalId) REFERENCES Person (nationalId) ON DELETE CASCADE,

CONSTRAINT nationalId\_check CHECK (nationalId ~ '^[0-9]{6}\/[0-9]{4}$')

);

CREATE TABLE IF NOT EXISTS Customer (

nationalId CHAR(11) PRIMARY KEY,

FOREIGN KEY (nationalId) REFERENCES Person (nationalId) ON DELETE CASCADE,

CONSTRAINT nationalId\_check CHECK (nationalId ~ '^[0-9]{6}\/[0-9]{4}$')

);

CREATE TABLE IF NOT EXISTS Trainer (

nationalId CHAR(11) PRIMARY KEY,

FOREIGN KEY (nationalId) REFERENCES Person (nationalId) ON DELETE CASCADE,

CONSTRAINT nationalId\_check CHECK (nationalId ~ '^[0-9]{6}\/[0-9]{4}$')

);

CREATE TABLE IF NOT EXISTS Supervise (

supervised CHAR(11) PRIMARY KEY,

supervisor CHAR(11) NOT NULL,

FOREIGN KEY (supervised) REFERENCES Employee(nationalId) ON DELETE CASCADE,

FOREIGN KEY (supervisor) REFERENCES Employee(nationalId) ON DELETE CASCADE,

CONSTRAINT supervised\_check CHECK (supervised ~ '^[0-9]{6}\/[0-9]{4}$'),

CONSTRAINT supervisor\_check CHECK (supervisor ~ '^[0-9]{6}\/[0-9]{4}$')

);

CREATE TABLE IF NOT EXISTS Invoice (

invoiceNumber INT PRIMARY KEY,

employee CHAR(11) NOT NULL,

dateOfIssue DATE DEFAULT now(),

cost DECIMAL(10, 2) NOT NULL,

FOREIGN KEY (employee) REFERENCES Employee (nationalId) ON DELETE SET NULL,

CONSTRAINT cost\_check CHECK (cost >= 0),

CONSTRAINT dateOfIssue\_check CHECK (dateOfIssue <= now()),

CONSTRAINT employee\_check CHECK (employee ~ '^[0-9]{6}\/[0-9]{4}$')

);

CREATE TABLE IF NOT EXISTS Membership (

type VARCHAR(120),

customer CHAR(11),

invoice INT,

expireDate Date NOT NULL,

cost DECIMAL(10, 2) DEFAULT 0,

PRIMARY KEY (type, customer, invoice),

FOREIGN KEY (customer) REFERENCES Customer (nationalId) ON DELETE CASCADE,

FOREIGN KEY (invoice) REFERENCES Invoice (invoiceNumber) ON DELETE CASCADE,

CONSTRAINT cost\_check CHECK (cost >= 0),

CONSTRAINT customer\_check CHECK (customer ~ '^[0-9]{6}\/[0-9]{4}$')

);

CREATE TABLE IF NOT EXISTS TrainersLicense (

trainer CHAR(11),

license VARCHAR(120),

PRIMARY KEY (trainer, license),

FOREIGN KEY (trainer) REFERENCES Trainer (nationalId) ON DELETE CASCADE,

CONSTRAINT trainer\_check CHECK (trainer ~ '^[0-9]{6}\/[0-9]{4}$')

);

CREATE TABLE IF NOT EXISTS WorkoutClass(

name VARCHAR(120),

day DATE,

time TIME,

teacher CHAR(11),

capacity INT,

PRIMARY KEY (name, day, time, teacher),

CONSTRAINT class\_capacity CHECK (capacity >= 0),

CONSTRAINT teacher\_fk

FOREIGN KEY (teacher) REFERENCES Trainer (nationalId)

ON DELETE SET NULL,

CONSTRAINT teacher\_check CHECK (teacher ~ '^[0-9]{6}\/[0-9]{4}$')

);

CREATE TABLE IF NOT EXISTS TakeClass(

customer CHAR(11),

name VARCHAR(120),

day DATE,

time TIME,

teacher CHAR(11),

price DECIMAL(10, 2) NOT NULL,

PRIMARY KEY (customer, name, day, time, teacher),

CONSTRAINT customer\_fk

FOREIGN KEY (customer) REFERENCES Customer (nationalId)

ON DELETE SET NULL,

CONSTRAINT teacher\_fk

FOREIGN KEY (teacher) REFERENCES Trainer (nationalId)

ON DELETE SET NULL,

CONSTRAINT check\_price CHECK (price >= 0),

CONSTRAINT customer\_check CHECK (customer ~ '^[0-9]{6}\/[0-9]{4}$'),

CONSTRAINT teacher\_check CHECK (teacher ~ '^[0-9]{6}\/[0-9]{4}$')

);

## SQL dotazy pro získání údajů z databáze

--- outer join ---

/\*

Joins table Person and Membership on same nationalid of person/customer and only person from city Hluk.

Result is rows selected by SELECT clause when they meet condition and also row where one of

conditions is missing.

\*/

SELECT person.firstname, person.lastname, person.city ,membership.type

FROM membership

FULL OUTER JOIN person

ON (person.nationalid = membership.customer)

WHERE (person.city = 'Hluk');

Table

Description automatically generated

--- inner join ---

/\*

Joins table Person and WorkoutClass on same nationalid of person/teacher. Result

contains only persons who match some teacher in workoutclasses. And where capacity < 10.

\*/

SELECT person.\*, workoutclass.name, workoutclass.capacity

FROM person

INNER JOIN workoutclass

ON (person.nationalid = workoutclass.teacher)

WHERE (workoutclass.capacity < 10);

Graphical user interface

Description automatically generated

--- condition on data ---

/\*

SELECT all workoutclasses that start after 12:00:00.

\*/

SELECT \* FROM workoutclass

WHERE (workoutclass."time" >= '12:00:00');

Table

Description automatically generated

--- agregaci a podmínku na hodnotu agregační funkce ---

/\*

Takes all types of classes and compute average price, which each customer paid for this type of class. Then

filter those having average > 12 and order them in acending order by average.

\*/

SELECT takeclass.name, AVG(takeclass.price) AS average

FROM takeclass

GROUP BY takeclass.name

HAVING AVG(takeclass.price) > 12

ORDER BY average;

Graphical user interface, application, Teams

Description automatically generated

--- řazení a stránkování ---

/\*

SELECT all membershiptypes only once and order them in ascendend order.

\*/

SELECT DISTINCT "type" FROM membership

ORDER BY "type" ASC;

Graphical user interface, application

Description automatically generated with medium confidence

--- množinové operace ---

/\*

Query return union of trainer and customer table with duplicities.

\*/

SELECT \* FROM trainer

UNION

SELECT \* FROM customer;

Graphical user interface, application

Description automatically generated

--- vnořený SELECT ---

/\*

Query returns table with first name and last name of employee from person table

and his position from employee table. At last it filter only those who has

over average salary.

\*/

SELECT p.firstName, p.lastName, e.workposition

FROM person AS p

JOIN employee AS e ON e.nationalid = p.nationalid

WHERE (e.salary > (SELECT AVG(salary) FROM employee));

Graphical user interface, application

Description automatically generated