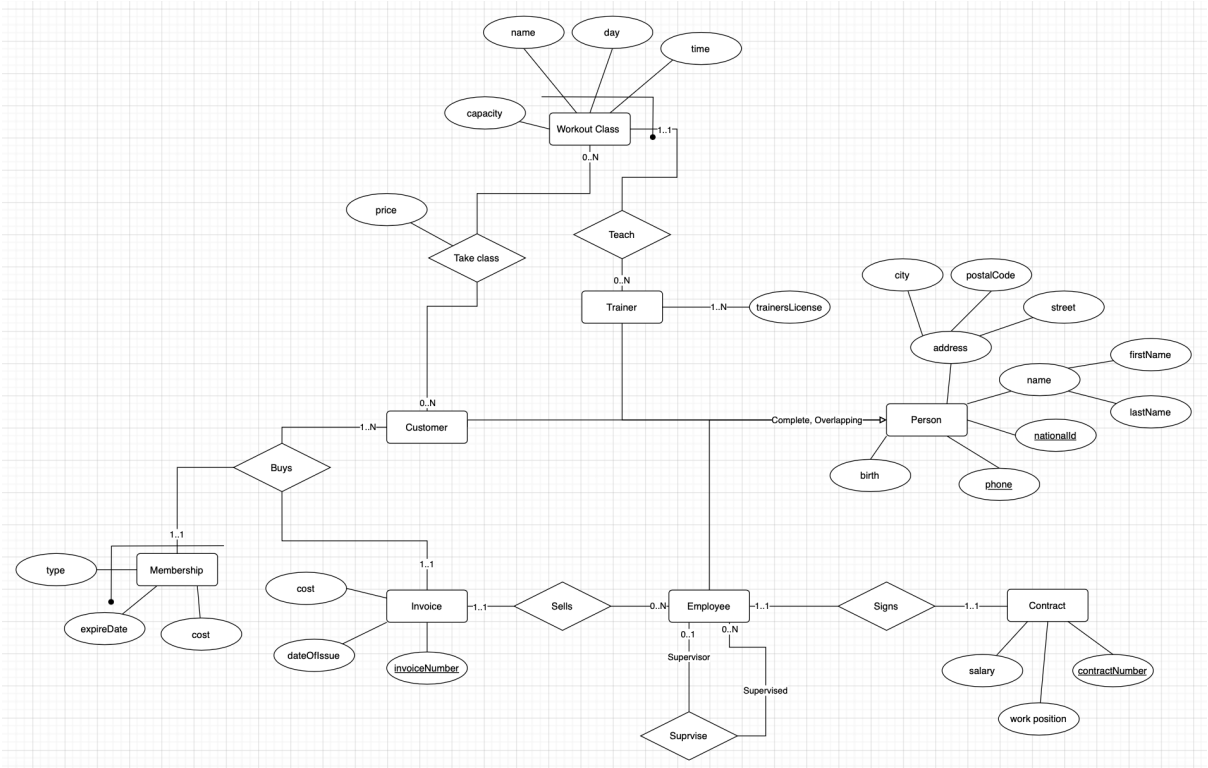
# Relační model

## Konceptuální model




---

**Person** (<u>nationalId</u>, <u>phone</u>, birth, firstName, lastName, city, street, postalCode)

**Employee** (<u>nationalId</u>, <u>contractNumber</u>, workPosition, salary)
FK: (naitonalNumber) ⊆ Person(nationalId)

**Supervise** (<u>supervised</u>, supervisor)
FK: (supervised) ⊆ Employee (nationalId)
FK: (supervisor) ⊆ Employee (nationalId)

**Invoice** (<u>invoiceNumber</u>, employee, dateOfIssue, cost)
FK: employee ⊆ Employee(nationalId)

**Customer** (<u>nationalId</u>)
FK: (naitonalNumber) ⊆ Person(nationalId)

**Membership** (<u>type, customer, invoice</u>, expireDate, cost)
FK: (customer) ⊆ Customer(nationalId)
FK: (invoice) ⊆ Invoice(invoiceNumber)

**Trainer** (<u>nationalId</u>)
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');
```

| | firstname character varying (120) | lastname character varying (120) | city character varying (120) | type character varying (120) |
|---|---|---|---|---|
| 1 | Jana | Vlčková | Hluk | Premium |
| 2 | René | Pokorná | Hluk | Premium |
| 3 | Ladislava | Kučera | Hluk | One Time |
| 4 | Vítězslav | Fialová | Hluk | Basic |
| 5 | Tadeáš | Dvořáková | Hluk | One Time |
| 6 | Ondřej | Novotný | Hluk | Basic |
| 7 | Božena | Urbanová | Hluk | One Time |
| 8 | Robert | Hájek | Hluk | Student |

--- 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);
```

| | nationalid character (11) | phone character (11) | birth date | firstname character varying (120) | lastname character varying (120) | city character varying (120) | street character vary |
|---|---|---|---|---|---|---|---|
| 1 | 291052/6063 | 734 091 868 | 1922-06-02 | Františka | Křížová | Osečná | Vizovická |
| 2 | 555341/7017 | 775 865 986 | 1994-03-14 | Bedřich | Kříž | Nalžovské Hory | Bermanova |
| 3 | 821270/0954 | 732 284 829 | 1979-09-22 | Miroslava | Doležal | Duchcov | U Smíchovsk |
| 4 | 375139/1882 | 777 117 358 | 1919-06-19 | Bohumila | Benešová | Bohušovice nad Ohří | Nábřeží Ludv |
| 5 | 780053/2819 | 774 558 642 | 1990-03-04 | Martina | Svobodová | Brušperk | Čerpadlová |
| 6 | 046357/9218 | 738 210 786 | 1996-08-31 | Hana | Soukupová | Chabařovice | Michnova |
| 7 | 874720/6491 | 775 285 573 | 2000-11-08 | Eva | Beranová | Špindlerův Mlýn | Malešovská |

--- condition on data ---

```
/*
SELECT all workoutclasses that start after 12:00:00.
```

```
*/
SELECT * FROM workoutclass
WHERE (workoutclass."time" >= '12:00:00');
```

| | name [PK] character varying (120) | day [PK] date | time [PK] time without time zone | teacher [PK] character (11) | capacity integer |
|---|---|---|---|---|---|
| 1 | CrossFit | 2024-01-16 | 22:00:00 | 525939/6852 | 9 |
| 2 | CrossFit | 2023-05-09 | 14:00:00 | 168906/4506 | 16 |
| 3 | Abs training | 2023-10-28 | 13:00:00 | 370902/7289 | 13 |
| 4 | Stretching | 2023-08-31 | 22:00:00 | 692295/7629 | 18 |
| 5 | PowerLifting | 2024-03-27 | 23:00:00 | 782165/5663 | 10 |
| 6 | Stretching | 2024-01-12 | 22:00:00 | 475915/5105 | 11 |
| 7 | Stretching | 2023-06-10 | 17:00:00 | 692209/7848 | 19 |
| 8 | Yoga | 2023-09-08 | 12:00:00 | 088434/2519 | 18 |

--- 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;
```

| | name character varying (120) | average numeric |
|---|---|---|
| 1 | Yoga | 12.4167749757516974 |
| 2 | Abs training | 12.4206101511879050 |
| 3 | CrossFit | 12.4434989316239316 |
| 4 | Zumba | 12.4814676486059968 |
| 5 | PowerLifting | 12.4849805636540330 |
| 6 | Power Yoga | 12.5184282238442822 |
| 7 | Pilates | 12.5289478718316595 |
| 8 | Stretching | 12.5384290460370195 |

--- řazení a stránkování ---

```
/*
SELECT all membershiptypes only once and order them in ascendend order.
*/
SELECT DISTINCT "type" FROM membership
ORDER BY "type" ASC;
```

| | type character varying (120) |
|---|---|
| 1 | 20 Times |
| 2 | Basic |
| 3 | One Time |
| 4 | Premium |
| 5 | Student |

--- množinové operace ---

```
/*
Query return union of trainer and customer table with duplicities.
*/
SELECT * FROM trainer
UNION
SELECT * FROM customer;
```

| | nationalid character (11) |
|---|---|
| 1 | 704550/2352 |
| 2 | 682389/7769 |
| 3 | 033614/3457 |
| 4 | 121143/6921 |
| 5 | 526172/7230 |
| 6 | 321536/9006 |
| 7 | 273445/5709 |
| 8 | 240256/7789 |

--- 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));
```

| | firstname character varying (120) | lastname character varying (120) | workposition character varying (120) |
|---|---|---|---|
| 1 | Jitka | Pospíšilová | Fyzioterapeut |
| 2 | Dominik | Vaňková | Food technologist |
| 3 | Stanislav | Čermák | Financial manager |
| 4 | Radim | Kučera | Financial manager |
| 5 | Veronika | Svobodová | Cleaner |
| 6 | Vladimír | Bartoš | Fyzioterapeut |
| 7 | Marie | Bartoš | Receptionist |
| 8 | Viktorie | Horáková | Immunologist |