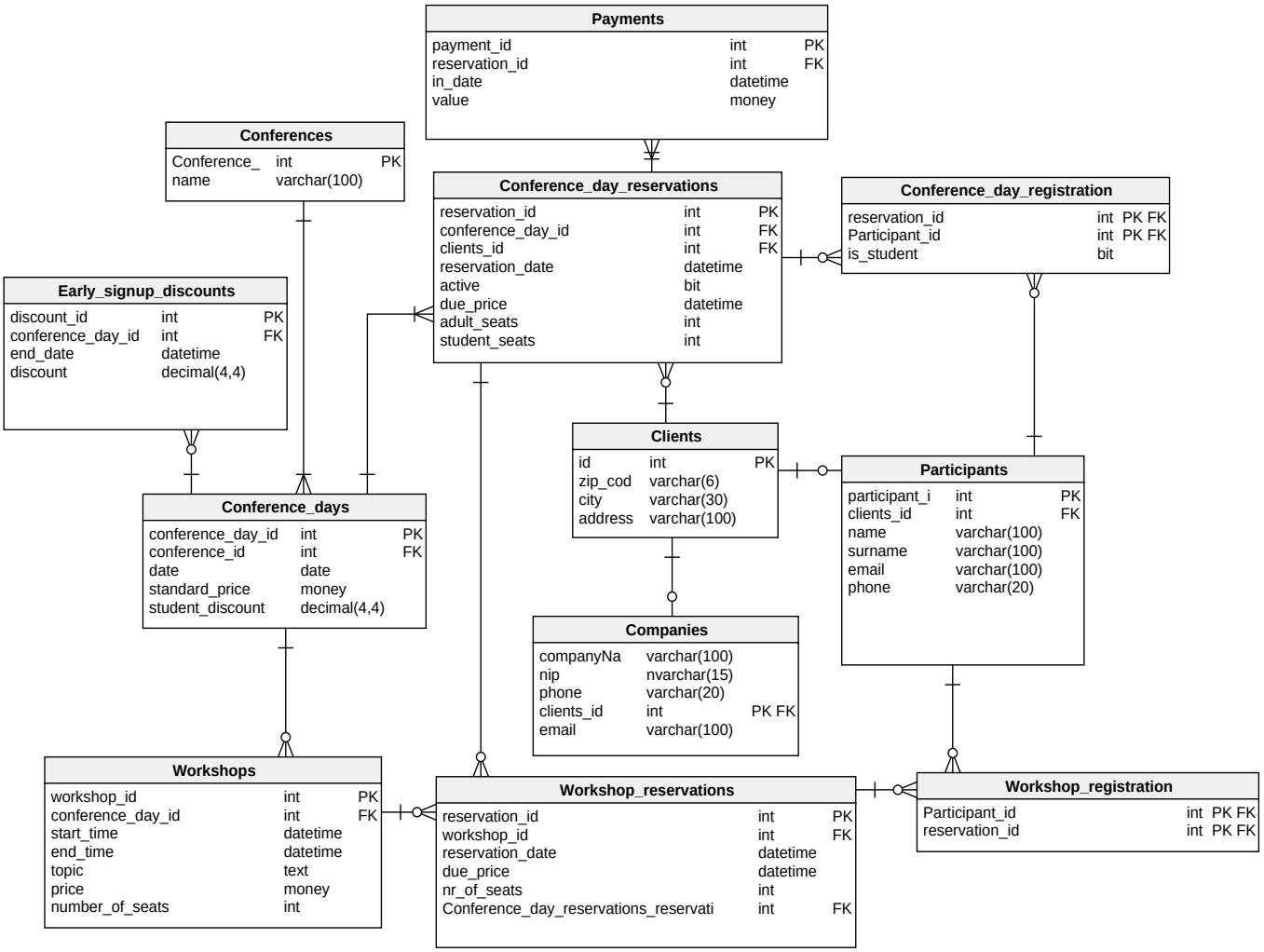
Podstawy baz danych
Projekt konferencje

Agnieszka Dutka, Maciek Trątnowiecki

AGH, Styczeń 2020

Objaśnienie schematu bazy

- Clients - Reprezentuje klientów chcących opłacić miejsca na konferencjach i warsztatach. Klientem może być zarówno firma, jak i osoba prywatna. W zależności od tego dane klienta reprezentowane są przez odpowiednią relację w bazie.
- Companies - Jeśli klient jest firmą, przechowuje jego dane.
- Participants - Jeśli klient jest osobą prywatną, przechowuje jego dane.
- Conferences - Reprezentuje konferencję z którą powiązane są odpowiednie dni konferencyjne, oraz warsztaty.
- Conference_days - Reprezentuje pojedynczy dzień konferencji. Powiązana jest z nim ustalona opłata za uczestnictwo. Zniżki obowiązujące w zależności od daty rejestracji zwarte są w relacji Early_Signup_Discounts.
- Early_Signup_Discounts - Odpowiada za informację o tabeli zniżek na dany dzień konferencyjny. Pojedyncza zniżka przechowywana jest w krotce z atrybutami w postaci procentowej obniżki ceny standardowej, oraz ostatniego dnia w którym obowiązuje.
- Conference_day_reservations - Realizuje rezerwacje na poszczególny dzień konferencji. Każda rezerwacja powiązana jest z klientem, który ją opłaca. Za powiązanie rezerwacji z uczestnikiem odpowiada osobna relacja. Zawiera także pole due_price określające termin płatności. Atrybut active odpowiada za możliwość rezygnacji z podjętej rezerwacji (uznaliśmy, że usuwanie krotki z bazy może nie być optymalnym rozwiązaniem, jako że zawarte w niej dane mogą jeszcze być przydatne z punktu widzenia logiki biznesowej). Atrybuty adult_seats i student_seats służą do liczenia kosztu podjęcia rezerwacji przed powiązaniem jej z uczestnikami konferencji.
- Conference_day_registration - Wiąże rezerwację z uczestnikami konferencji. Atrybut is_student informuje, czy danemu uczestnikowi przysługuje zniżka studencka.
- Payments - Przechowuje informacje o wpływach pieniężnych powiązanych z daną rejestracją.
- Workshops - Reprezentuje warsztaty odbywające się w trakcie odpowiednich dni konferencyjnych.
- Workshops_reservations - Opisuje rezerwacje na warsztaty w sposób analogiczny do rezerwacji na konferencje.
- Workshops_registrations - Łączy rezerwację z uczestnikami w sposób analogiczny do dni konferencyjnych.


Payments
payment_id int PK
reservation_id int FK
in_date datetime
value money
Conferences
Conference_ int PK
name varchar(100)
Conference_day_reservations
reservation_id int PK
conference_day_id int FK
clients_id int FK
reservation_date datetime
active bit
due_price datetime
adult_seats int
student_seats int
Conference_day_registration
reservation_id int PK FK
Participant_id int PK FK
is_student bit
Early_signup_discounts
discount_id int PK
conference_day_id int FK
end_date datetime
discount decimal(4,4)
Clients
id int PK
zip_cod varchar(6)
city varchar(30)
address varchar(100)
Participants
participant_i int PK
clients_id int FK
name varchar(100)
surname varchar(100)
email varchar(100)
phone varchar(20)
Conference_days
conference_day_id int PK
conference_id int FK
date date
standard_price money
student_discount decimal(4,4)
Companies
companyNa varchar(100)
nip nvarchar(15)
phone varchar(20)
clients_id int PK FK
email varchar(100)
Workshops
workshop_id int PK
conference_day_id int FK
start_time datetime
end_time datetime
topic text
price money
number_of_seats int
Workshop_reservations
reservation_id int PK
workshop_id int FK
reservation_date datetime
due_price datetime
nr_of_seats int
Conference_day_reservations_reservati int FK
Workshop_registration
Participant_id int PK FK
reservation_id int PK FK

Implementacja

```
-- tables
-- Table: Clients
CREATE TABLE Clients (
    id int  NOT NULL IDENTITY,
    zip_code varchar(6)  NOT NULL CHECK (zip_code like '[0-9][0-9]-[0-9][0-9][0-9]' ),
    city varchar(30)  NOT NULL CHECK (city not like '%[0-9]%'),
    address varchar(100)  NOT NULL,
    CONSTRAINT Clients_pk PRIMARY KEY  (id)
);

-- Table: Companies
CREATE TABLE Companies (
    companyName varchar(100)  NOT NULL,
    nip nvarchar(15)  NOT NULL CHECK ((nip not like '%[^0-9]%') and (LEN(nip) = 10) and (nip not like '0%' or nip like '1%')),
    phone varchar(20)  NOT NULL CHECK (phone not like '%[^0-9]%'),
    clients_id int  NOT NULL,
    email varchar(100)  NOT NULL CHECK (email like '%_@__%.__%'),
    CONSTRAINT unique_nip UNIQUE (nip),
    CONSTRAINT checkNip CHECK (dbo.IsValidNip(nip) = 1),
    CONSTRAINT Companies_pk PRIMARY KEY  (clients_id)
);

-- Table: Conference_day_registration
CREATE TABLE Conference_day_registration (
    reservation_id int  NOT NULL,
    Participant_id int  NOT NULL,
    is_student bit  NOT NULL DEFAULT 0,
    CONSTRAINT Conference_day_registration_pk PRIMARY KEY  (reservation_id,Participant_id)
);

-- Table: Conference_day_reservations
CREATE TABLE Conference_day_reservations (
    reservation_id int  NOT NULL IDENTITY,
    conference_day_id int  NOT NULL,
    clients_id int  NOT NULL,
    reservation_date datetime  NOT NULL DEFAULT GETDATE(),
    active bit  NOT NULL DEFAULT 1,
    due_price datetime  NOT NULL DEFAULT DATEADD(week, 2, GETDATE()) CHECK (due_price >= GETDATE()),
    adult_seats int  NOT NULL DEFAULT 0 CHECK (adult_seats >= 0),
    student_seats int  NOT NULL DEFAULT 0 CHECK (student_seats >= 0),
    CONSTRAINT Conference_day_reservations_pk PRIMARY KEY  (reservation_id)
);

-- Table: Conference_days
CREATE TABLE Conference_days (
    conference_day_id int  NOT NULL IDENTITY,
    conference_id int  NOT NULL,
    date date  NOT NULL DEFAULT GETDATE(),
    standard_price money  NOT NULL DEFAULT 0 CHECK (standard_price >= 0),
    student_discount decimal(4,4)  NOT NULL DEFAULT 0 CHECK (student_discount >= 0),
    CONSTRAINT Conference_days_pk PRIMARY KEY  (conference_day_id)
);

-- Table: Conferences
CREATE TABLE Conferences (
    Conference_id int  NOT NULL IDENTITY,
    name varchar(100)  NOT NULL,
    CONSTRAINT Conferences_pk PRIMARY KEY  (Conference_id)
);

-- Table: Early_signup_discounts
CREATE TABLE Early_signup_discounts (
    discount_id int  NOT NULL IDENTITY,
    conference_day_id int  NOT NULL,
    end_date datetime  NOT NULL,
    discount decimal(4,4)  NOT NULL DEFAULT 0 CHECK (discount >= 0),
    CONSTRAINT Early_signup_discounts_pk PRIMARY KEY  (discount_id)
);

-- Table: Participants
CREATE TABLE Participants (
    participant_id int  NOT NULL IDENTITY,
```

```
    clients_id int  NOT NULL,
    name varchar(100)  NOT NULL CHECK (name like '^\p{L}+$'),
    surname varchar(100)  NOT NULL CHECK (surname like '^\p{L}+$'),
    email varchar(100)  NOT NULL CHECK (email like '%_@__%.__%'),
    phone varchar(20)  NOT NULL CHECK (phone not like '%[^0-9]%'),
    CONSTRAINT Participants_pk PRIMARY KEY  (participant_id)
);

-- Table: Payments
CREATE TABLE Payments (
    payment_id int  NOT NULL IDENTITY,
    reservation_id int  NOT NULL,
    in_date datetime  NOT NULL,
    value money  NOT NULL,
    CONSTRAINT Payments_pk PRIMARY KEY  (payment_id)
);

-- Table: Workshop_registration
CREATE TABLE Workshop_registration (
    Participant_id int  NOT NULL,
    reservation_id int  NOT NULL,
    CONSTRAINT Workshop_registration_pk PRIMARY KEY  (Participant_id,reservation_id)
);

-- Table: Workshop_reservations
CREATE TABLE Workshop_reservations (
    reservation_id int  NOT NULL IDENTITY,
    workshop_id int  NOT NULL,
    reservation_date datetime  NOT NULL DEFAULT GETDATE(),
    due_price datetime  NOT NULL DEFAULT DATEADD(week, 2, GETDATE()) CHECK (due_price >=
    GETDATE()),
    nr_of_seats int  NOT NULL DEFAULT 0 CHECK (nr_of_seats >= 0),
    Conference_day_reservations_reservation_id int  NOT NULL,
    CONSTRAINT Workshop_reservations_pk PRIMARY KEY  (reservation_id)
);

-- Table: Workshops
CREATE TABLE Workshops (
    workshop_id int  NOT NULL IDENTITY,
    conference_day_id int  NOT NULL,
    start_time datetime  NOT NULL,
    end_time datetime  NOT NULL CHECK (end_time >= GETDATE()),
    topic text  NOT NULL,
    price money  NOT NULL CHECK (price >= 0),
    number_of_seats int  NOT NULL DEFAULT 0 CHECK (number_of_seats >= 0),
    CONSTRAINT Workshops_pk PRIMARY KEY  (workshop_id)
);

-- foreign keys
-- Reference: Companies_Clients (table: Companies)
ALTER TABLE Companies ADD CONSTRAINT Companies_Clients
    FOREIGN KEY (clients_id)
    REFERENCES Clients (id);

-- Reference: Conference_day_registration_Conference_day_reservations (table:
    Conference_day_registration)
ALTER TABLE Conference_day_registration ADD CONSTRAINT
    Conference_day_registration_Conference_day_reservations
    FOREIGN KEY (reservation_id)
    REFERENCES Conference_day_reservations (reservation_id);

-- Reference: Conference_day_registration_Participants (table: Conference_day_registration)
ALTER TABLE Conference_day_registration ADD CONSTRAINT
    Conference_day_registration_Participants
    FOREIGN KEY (Participant_id)
    REFERENCES Participants (participant_id);

-- Reference: Conference_day_reservations_Clients (table: Conference_day_reservations)
ALTER TABLE Conference_day_reservations ADD CONSTRAINT Conference_day_reservations_Clients
    FOREIGN KEY (clients_id)
    REFERENCES Clients (id);

-- Reference: Conference_day_reservations_Conference_days (table: Conference_day_reservations)
ALTER TABLE Conference_day_reservations ADD CONSTRAINT
    Conference_day_reservations_Conference_days
    FOREIGN KEY (conference_day_id)
```

```
    REFERENCES Conference_days (conference_day_id);

-- Reference: Conference_days_Conferences (table: Conference_days)
ALTER TABLE Conference_days ADD CONSTRAINT Conference_days_Conferences
    FOREIGN KEY (conference_id)
    REFERENCES Conferences (Conference_id);

-- Reference: Discounts_Conference_days (table: Early_signup_discounts)
ALTER TABLE Early_signup_discounts ADD CONSTRAINT Discounts_Conference_days
    FOREIGN KEY (conference_day_id)
    REFERENCES Conference_days (conference_day_id);

-- Reference: Participants_Clients (table: Participants)
ALTER TABLE Participants ADD CONSTRAINT Participants_Clients
    FOREIGN KEY (clients_id)
    REFERENCES Clients (id);

-- Reference: Payments_Conference_day_reservations (table: Payments)
ALTER TABLE Payments ADD CONSTRAINT Payments_Conference_day_reservations
    FOREIGN KEY (reservation_id)
    REFERENCES Conference_day_reservations (reservation_id);

-- Reference: Workshop_registration_Participants (table: Workshop_registration)
ALTER TABLE Workshop_registration ADD CONSTRAINT Workshop_registration_Participants
    FOREIGN KEY (Participant_id)
    REFERENCES Participants (participant_id);

-- Reference: Workshop_registration_Workshop_reservations (table: Workshop_registration)
ALTER TABLE Workshop_registration ADD CONSTRAINT Workshop_registration_Workshop_reservations
    FOREIGN KEY (reservation_id)
    REFERENCES Workshop_reservations (reservation_id);

-- Reference: Workshop_reservations_Conference_day_reservations (table: Workshop_reservations)
ALTER TABLE Workshop_reservations ADD CONSTRAINT
    Workshop_reservations_Conference_day_reservations
    FOREIGN KEY (Conference_day_reservations_reservation_id)
    REFERENCES Conference_day_reservations (reservation_id);

-- Reference: Workshop_reservations_Workshops (table: Workshop_reservations)
ALTER TABLE Workshop_reservations ADD CONSTRAINT Workshop_reservations_Workshops
    FOREIGN KEY (workshop_id)
    REFERENCES Workshops (workshop_id);

-- Reference: Workshops_Conference_days (table: Workshops)
ALTER TABLE Workshops ADD CONSTRAINT Workshops_Conference_days
    FOREIGN KEY (conference_day_id)
    REFERENCES Conference_days (conference_day_id);

-- End of file.
```

../Create.sql

```
1000 CREATE FUNCTION IsValidNip
     (
1002   @nip nvarchar(15)
     )
1004 RETURNS bit
     AS
1006 BEGIN
         IF ISNUMERIC(@nip) = 0 BEGIN
1008         RETURN 0
             END
1010
         IF @nip='0000000000' BEGIN
1012         RETURN 0
             END
1014     IF @nip='1234567891' BEGIN
             RETURN 0
1016         END
         IF @nip='1111111111' BEGIN
1018         RETURN 0
             END
1020     IF @nip='1111111112' BEGIN
             RETURN 0
1022         END
         IF @nip='9999999999' BEGIN
1024         RETURN 0
             END
1026     IF @nip='1111111112' BEGIN
             RETURN 0
1028         END

1030     DECLARE @sum TINYINT;
         SET @sum = 6 * CONVERT(TINYINT, SUBSTRING(@nip,1,1)) +
1032                5 * CONVERT(TINYINT, SUBSTRING(@nip,2,1)) +
                    7 * CONVERT(TINYINT, SUBSTRING(@nip,3,1)) +
1034                2 * CONVERT(TINYINT, SUBSTRING(@nip,4,1)) +
                    3 * CONVERT(TINYINT, SUBSTRING(@nip,5,1)) +
1036                4 * CONVERT(TINYINT, SUBSTRING(@nip,6,1)) +
                    5 * CONVERT(TINYINT, SUBSTRING(@nip,7,1)) +
1038                6 * CONVERT(TINYINT, SUBSTRING(@nip,8,1)) +
                    7 * CONVERT(TINYINT, SUBSTRING(@nip,9,1));
1040
         IF CONVERT(TINYINT,SUBSTRING(@nip,10,1)) = (@sum % 11) BEGIN
1042         RETURN 1
             END
1044     RETURN 0
     END
```

../checkNip.sql