Database Design

Współczesne zastosowania informatyki – zadanie 6

Jakub Nadolski, Igor Gula

Użytkownicy

```sql
CREATE TABLE Uzytkownicy (
  id_uzytkownika INTEGER PRIMARY KEY NOT NULL,
  imie VARCHAR(45) NOT NULL,
  nazwisko VARCHAR(45) NOT NULL,
  email VARCHAR(45) NOT NULL,
  haslo VARCHAR(45) NOT NULL,
  data_urodzenia DATE CHECK(AGE(data_urodzenia) >= INTERVAL '13 years')
);
```

W tabeli Użytkownicy mamy wszystkie najważniejsze dane użytkowników i sprawdzamy czy mają co najmniej 13 lat.

Adresy

```sql
CREATE TABLE adresy (
  id_adresu INTEGER PRIMARY KEY NOT NULL,
  miejscowosc VARCHAR(45),
  kod_pocztowy VARCHAR(5) NOT NULL CHECK(CHAR_LENGTH(kod_pocztowy) = 5),
  ulica VARCHAR(45) NOT NULL,
  numer_budynku INTEGER NOT NULL,
  numer_lokalu INTEGER
);
```

W tabeli Adresy sprawdzamy czy kod pocztowy ma odpowiednią długość.

Lokacje

```
CREATE TABLE Lokacje (
  id_lokacji INTEGER PRIMARY KEY NOT NULL,
  nazwa_lokacji VARCHAR(45) NOT NULL,
  rodzaj_lokacji VARCHAR(45) NOT NULL CHECK(rodzaj_lokacji IN ('nocleg', 'atrakcja')),
  id_adresu INTEGER NOT NULL REFERENCES adresy(id_adresu),
  id_wlasciciela integer NOT NULL REFERENCES uzytkownicy(id_uzytkownika),
  opis_lokacji TEXT,
  kontakt_telefon VARCHAR(9) NOT NULL CHECK(CHAR_LENGTH(kontakt_telefon) = 9),
  kontakt_email VARCHAR(45) NOT NULL
 );
```

W tabeli Lokacje mamy nazwę oraz rodzaj lokacji, jej adres, właściciela, kontakt do niego i opis lokacji. Sprawdzamy czy rodzaj lokacji jest jednym z dwóch podanych oraz czy numer telefonu ma długość 9.

Rezerwacje

```sql
CREATE TABLE Rezerwacje (
 id_rezerwacji INTEGER PRIMARY KEY NOT NULL,
 id_uzytkownika INTEGER NOT NULL REFERENCES uzytkownicy(id_uzytkownika),
 data_poczatku_rezerwacji DATE NOT NULL,
 data_konca_rezerwacji DATE NOT NULL,
 id_lokacji INTEGER NOT NULL REFERENCES lokacje(id_lokacji),
 cena_rezerwacji FLOAT CHECK(cena_rezerwacji > 0.00),
 status_rezerwacji VARCHAR(45) NOT NULL CHECK(status_rezerwacji IN ('potwierdzona', 'anulowana', 'nieznany'))
);
```

W tabeli Rezerwacje mamy użytkownika oraz rezerwowaną lokację, daty początku i końca rezerwacji, jej cenę i status. Sprawdzamy czy cena rezerwacji jest większa od 0 oraz czy status rezerwacji jest jednym z 3 prawidłowych statusów.

Opinie

```sql
CREATE TABLE Opinie (
  id_opinii INTEGER PRIMARY KEY NOT NULL,
  id_uzytkownika INTEGER NOT NULL REFERENCES uzytkownicy(id_uzytkownika),
  id_lokacji INTEGER NOT NULL REFERENCES lokacje(id_lokacji),
  data_dodania DATE NOT NULL,
  ocena_lokacji INTEGER NOT NULL CHECK(ocena_lokacji BETWEEN 1 AND 10),
  tresc_opinii TEXT
);
```

W tabeli Opinie znajduje się id użytkownika i lokacji, data wystawienia opinii i ocena lokacji. Sprawdzamy czy ocena lokacji wynosi od 1 do 10.

Zdjęcia

```sql
CREATE TABLE Zdjecia (
  id_zdjecia INTEGER PRIMARY KEY NOT NULL,
  id_lokacji INTEGER NOT NULL REFERENCES lokacje(id_lokacji),
  link_do_zdjecia VARCHAR(255) NOT NULL
);
```

Tworzymy tabelę zdjęcia, w której umieszczamy link do zdjęcia lokacji

Grupy odbiorców

```sql
CREATE TABLE Grupy_odbiorcow (
  id_grupy_odbiorcow INTEGER PRIMARY KEY NOT NULL,
  nazwa_grupy VARCHAR(45) NOT NULL
);

CREATE TABLE Przypisanie_grupa_odbiorcow_lokacja (
  id_przypisania INTEGER PRIMARY KEY NOT NULL,
  id_lokacji INTEGER NOT NULL REFERENCES lokacje(id_lokacji),
  id_grupy_odbiorcow INTEGER NOT NULL REFERENCES grupy_odbiorcow(id_grupy_odbiorcow)
);
```

Tworzymy tabele Grupy odbiorców i Przypisanie grupy do lokacji.

Tagi

```sql
CREATE TABLE Tagi (
  id_tagu INTEGER PRIMARY KEY NOT NULL,
  nazwa_tagu VARCHAR(45) NOT NULL
);

CREATE TABLE Przypisanie_tagi_lokacja (
  id_przypisania INTEGER PRIMARY KEY NOT NULL,
  id_lokacji INTEGER NOT NULL REFERENCES lokacje(id_lokacji),
  id_tagu INTEGER NOT NULL REFERENCES tagi(id_tagu)
);
```

Tworzymy tabele Tagi i Przypisanie tagu do lokacji.

Plany podróży

```sql
CREATE TABLE Plany_podrozy (
  id_planu_podrozy INTEGER PRIMARY KEY NOT NULL,
  id_uzytkownika INTEGER NOT NULL REFERENCES uzytkownicy(id_uzytkownika)
);
```

W tabeli Plany podróży mamy tylko id_użytkownika, który utworzył ten plan.

Punkty planów

```sql
CREATE TABLE Punkty_planow (
  id_punktu_planu INTEGER PRIMARY KEY NOT NULL,
  id_planu_podrozy INTEGER NOT NULL REFERENCES plany_podrozy(id_planu_podrozy),
  id_lokacji INTEGER NOT NULL REFERENCES lokacje(id_lokacji),
  dzien_przybycia DATE,
  godzina_przybycia TIME,
  koszt INTEGER
);
```

W tabeli Punkty planów zawarte są szczegółowe informacje o konkretnym momencie w podróży, czyli data i godzina przybycia do konkretnej lokacji.