

Modelovanje baza podataka

Autori:
Milan Segedinac
Goran Savić

1. Modelovanje baza podataka

U prethodnim lekcijama videli smo da se baza podataka sastoji od tabela i da su tabele povezane spoljnim ključevima. Takođe, videli smo da je prvi korak u radu sa bazom podataka kreiranje tabela koje je uključivalo specifikaciju kolona. U realnim aplikacijama broj tabela je često velik (nekoliko stotina) i bilo bi teško da se direktno napišu CREATE naredbe za tabele, ako prethodno nije dizajniran sistem (slično kao što bi bilo teško direktno implementirati algoritam ako prvo nije dizajniran). Stoga koristimo neki od grafičkih jezika za modelovanje baza podataka, kao što smo dijagram klasa koristili za objektno modelovanje. Modelovanje skladišta podataka je jedan od prvih koraka u implementaciji softverskih proizvoda.

Model podataka tabele baze podataka predstavljamo kao skup međusobno povezanih entiteta („pravougaonika“) sa atributima. Jedan primer modela podataka dat je slikom ispod.



Slika – Model baze podataka

Intuitivno, možemo da pretpostavimo da se radi o bazi podataka koja ima dve tabele: *student* i *studijski_program*. Tabela *student* ima kolonu *id* koja je tipa *INT* u kojoj se nalaze primarni ključevi; kolone *ime* i *prezime* koje su obe tipa *VARCHAR* od 50 karaktera; kolonu *broj_indeksa* koja je tipa *VARCHAR* od 30 karaktera; i kolonu *studijski_program_id* koja je tipa *INT* u koju se smeštaju spoljni ključevi na slogove iz kolone *studijski_program*. Tabela *studijski_program* ima kolonu *id* tipa *INT* u koju su smešteni primarni ključevi; i kolonu *naziv* koja je tipa *VARCHAR* dužine 50 karaktera. SQL kod za kreiranje takve baze dat je listingom ispod.

```
CREATE SCHEMA IF NOT EXISTS mydb;
USE mydb ;

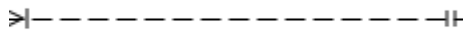
CREATE TABLE IF NOT EXISTS studijski_program (
  id INT NOT NULL AUTO_INCREMENT,
  naziv VARCHAR(50) NULL,
  PRIMARY KEY (id));

CREATE TABLE IF NOT EXISTS student (
  id INT NOT NULL AUTO_INCREMENT,
  ime VARCHAR(50) NULL,
  prezime VARCHAR(50) NULL,
  broj_indeksa VARCHAR(30) NULL,
  studijski_program_id INT NOT NULL,
  PRIMARY KEY (id),
  INDEX fk_student_studijski_program_idx (studijski_program_id ASC),
  CONSTRAINT fk_student_studijski_program
    FOREIGN KEY (studijski_program_id)
    REFERENCES studijski_program (id));
```

Slika – SQL skript za kreiranje baze

Jedan slog tabele *student* imaće najviše jedan identifikator sloga iz tabele *studijski_program*, što znači da će naš model podataka omogućiti da se student nalazi na najviše jednom studijskom

programu. Sa druge strane, model dozvoljava da više studenata bude upisano na jedan isti studijski program – prosto će više slogova u tabeli *student* u polju *studijski_program_id* imati kao vrednost *id* istog sloga iz tabele *studijski_program*. Ovakvu vezu zovemo *jedan-na-više* veza (1:n) i predstavljena je grafičkim simbolom sa slike ispod.



Slika – veza jedan-na-više

2. Nivoi modelovanja baza podataka

Postoje različiti nivoi modelovanja baza podataka. Najapstraktniji je *konceptualni model* u kom se navode samo *entitete sa svojim nazivima* i *veze*. Na nižem nivou apstrakcije je *logički model* u kom se navode još i *atributi* i specificiraju *primarni* i *spoljni* ključevi. Najkonkretniji je *fizički model podataka* u kome se navode i *tipovi*. Model podataka koji smo mi posmatrali u prethodnoj sekciji je *fizički model podataka*.






Modeli podataka na svim nivoima apstrakcije mogu se zadati u različitim notacijama. Obzirom da ćemo mi za modelovanje baze podataka koristiti MySQL Workbench, koristićemo i notaciju koja se koristi u ovom alatu. Stoga ćemo je ukratko objasniti. U ovoj notaciji *tabele* zadajemo kao *entitete*, a *kolone* kao *attribute entiteta*. Za svaku *tabelu* možemo da zadamo *naziv*, a za svaku kolonu tabele možemo da zadamo:

1. Naziv kolone
2. Tip
3. Default vrednost koja će biti postavljena ukoliko sami ne postavimo vrednost
4. Boolean oznake kojima se označava:
 - a. Da li je polje privatni ključ
 - b. Da li može da ima vrednost null
 - c. Da li je vrednost polja jedinstvena
 - d. Da li je vrednost binarna
 - e. Da li je vrednost neoznačena
 - f. Da li da se postavi 0 za brožčane vrednosti ukoliko nije uneta vrednost
 - g. Da li je vrednost auto incremental

Prema kardinalitetu imamo tri tipa veza:

1. Jedan-na-jedan (1:1) – primarni ključ jedne tabele postavljen je kao spoljni ključ u drugu tabelu
2. Jedan-na-više (1:n) – primarni ključ tabele sa strane '1' postavljen je kao spoljni ključ u tabelu na strani 'n'
3. Više-na-više (n:m) – u ovom slučaju kreira se nova (vezna) tabela. Primarni ključ u ovoj tabeli je kompozitni i sačinjen je od primarnih ključeva iz povezanih tabela.

Veze, pored podele po kardinalitetu, mogu da se podele i na *identifikujuće* (ukoliko spoljni ključ učestvuje u identifikovanju sloga, odnosno ako je deo primarnog ključa, na primer slog u tabeli *ispit* bi mogao da bude identifikovan vezom sa slogom u tabeli *student* i vezom sa slogom u tabeli *predmet*) i *neidentifikujuće* (ukoliko spoljni ključ ne učestvuje u identifikovanju sloga, na primer ukoliko se koristi surogat ključ). Obzirom da su više-na-više veze uvek identifikujuće (primarni ključ u veznoj tabeli uvek je kompozitni i sačinjen je od primarnih ključeva iz povezanih tabela) imamo pet tipova veza:

1.  - Neidentifikujuća jedan-na-jedan veza
2.  - Neidentifikujuća jedan-na-više veza
3.  - Identifikujuća jedan-na-jedan veza
4.  - Identifikujuća jedan-na-više veza
5.  - (Identifikujuća) više-na-više veza

Ovako kreirani model može se iskoristiti za generisanje SQL skripta za kreiranje tabela koje imaju:

1. *Nazive* koji odgovaraju *nazivima entiteta*;
2. *Nazive i tipove kolona* koji odgovaraju *nazivima i tipovima atributa*;
3. *Spoljne ključeve* u skladu sa vezama između entiteta.