Programtervezési minták értelmezése normálformákként

Kusper Gábor, gkusper@aries.ektf.hu
Eszterházy Károly Főiskola

Tartalomjegyzék

- Pár szóban a programtervezési mintákról
- A Relációs Adatmodell és az Objektum Orientált Programozás
- Normálformák a Relációs Adatmodellben
- Az OOP Normálformák következményei

Pár szóban a programtervezési mintákról

- A 90-es évektől kezdve kerülnek középpontba
- „Recept” a programozó számára
- Probléma viszont, hogy nincs egy egységesen elfogadott, oktatásban is jól használható modell

A Relációs Adatmodell és az Objektum Orientált Programozás

RA és OOP fogalmai közti megfeleltetés:

Tábla	⟺	Osztály
Attribútum	⟺	Metódus
Egyed	⟺	Objektum
Elsődleges kulcs	⟺	this
Idegen kulcs	⟺	Referencia

Normálformák a Relációs Adatmodellben

- Bár több normálformát ismerünk, de ezek közül a gyakorlatban az első három használata terjedt el:
  - 1NF: Egy reláció első normálformában van, ha a reláció minden értéke elemi.
  - 2NF: Egy reláció második normálformában van, ha első normálformájú, továbbá egyetlen másodlagos attribútuma sem függ funkcionálisan egyetlen kulcsának valódi részhalmazától sem.
  - 3NF: Egy reláció második normálformában van, ha első normálformájú, továbbá a másodlagos attribútumai közt nincs funkcionális függőség.

Normálformák a Relációs Adatmodellben

- Nem lehet azonban egyszerűen csak átírni a normálformákat ezen megfeleltetés alapján:
  - Már az első normálforma ilyen módú használatánál felmerülne az összetett típusok kérdése
  - Másrészt a normálformák az egyedek belső szerkezetére tesznek megkötést, de az objektumok belső állapota rejtve van.
- Ezért az OOP NF-eknek arról kell rendelkeznie, hogy egy objektum hogyan viszonyuljon a környezetéhez. Az OOP NF-eknek biztosítaniuk kell, hogy az objektumok kevéssé „csatoltak” legyenek.

Csatoltság

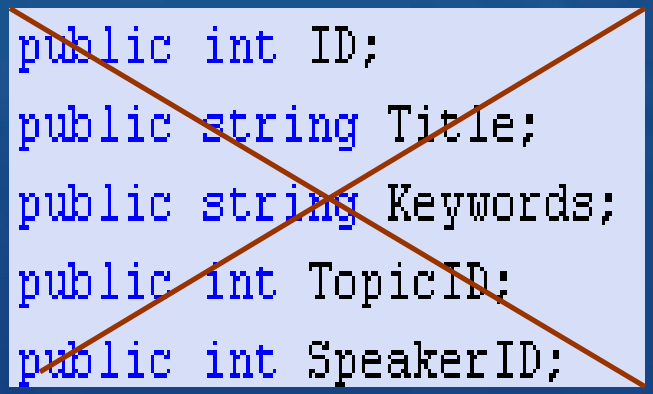
- Definíció (Larry Constantine alapján): A csatolás annak mértéke milyen erős kapcsolatban áll egy osztály a többi osztállyal. A csatolás mértéke két osztály *A* és *B* között növekszik, ha:
  - *A*-nak van *B* típusú mezője.
  - *A* meghívja *B* valamelyik metódusát.
  - *A*-nak van olyan metódusa, amelynek visszatérési típusa *B*.
  - *A* *B*-nek leszármazottja, vagy implementálja *B*-t.

Normálformák a Relációs Adatmodellben

- Ezek alapján a következő OOP normálformákat lehet javasolni:
  - 1NF: Az osztály első normálformában van, ha környezete előlről elrejti a mezőit.
  - 2NF: Az osztály második normálformában van, ha első normálformájú, továbbá a referencia típusú mezői elvont osztály típusúak.
  - 3NF: Az osztály harmadik normálformában van, ha második normálformájú, továbbá minden metódus hívás által visszaadott érték és belső állapot átmenet a környezetétől független.

Az OOP Normálformák következményei

- 1NF: Az egységbezárás elvének következménye, hogy a mezőket csak az objektum saját metódusai írhatják, olvashatják. De ennek betartása nem kötelező, egy mező akár publikus is lehet. Ezt zárja ki az első normálforma.



Az OOP Normálformák következményei

- 2NF:

  Tulajdonképpen a programtervezési minták első alapelvét fogalmazza meg, hiszen ha csak elvont osztályokat ismerünk, akkor csak a felületet ismerhetjük.

„Programozzunk a felületre a megvalósítás helyett!”

Az OOP Normálformák következményei

- 3NF: ha egy objektum metódusát meghívjuk. akkor az egyrészt visszaad egy értéket, másrészt megváltoztathatja az objektum belső állapotát. Ha a visszatérési érték és az állapot átmenet is csak az objektum belső állapotától és a metódus paramétereitől függ, akkor harmadik normálformában van az osztály.

Összefoglalásként

- Ahhoz, hogy a programtervezési minták leírását egyszerűsítsük, egy könnyebben formalizálható modellre van szükségünk
- Ebben a megközelítésben a RA és az OOP fogalmai között tettünk megfeleltetést
- Programtervezési minták leírása OOP NF-ekkel

Irodalomjegyzék

- S. W. Ambler: *Mapping Objects to a Relational Database*.
- S. W. Ambler és B. McGibbon: *Building Object Applications That Work.* Cambridge University Press/SIGS Books, 1997, ISBN: 0521-64826-2.
- L. Constantine, B. Hendersons-Sellers és I. M. Graham: *Coupling and Cohesion: Towards a Valid metrics Suite for Object-oriented Analysis and Design.* Object Oriented Systems, 3:143-158, 1996.
- The Gang of Four: E. Gamma, R. Helm, R. Johnson és J. Vlissides: *Design Patterns: Elements of Reusable Object-Oriented Software.* Pearson Education Inc., Addison Wesley Professional, 1995, ISBN: 0201-63361-2.
- Szabolcs M. és Kusper G.: *Understanding Design Patterns as Constructive Proofs.* Proceedings of ICAI 2004, Volume II., 173-182, 2004.
- J. Rumbaugh: *The life of an object model: How the object model changes during development.* J. of Object-Oriented Programming, 7(1):24-32, 1994.
- A. Weinand, E. Gamma és R. Marty: *ET++ - An object-oriented application framework in C++.* Object-Oriented Programming Systems, Languages and Applications Conference Proceedings, 46-57, 1988.

Köszönöm a figyelmet!