Vývoj informačních systémů

Vzory: Mapování dědičnosti

2023-24

Objektově-relační struktury

- *Identity Field*: Saves a database ID field in an object to maintain identity between an in-memory object and a database row.
- *Foreign Key Mapping*: Maps an association between objects to a foreign key reference between tables.
- *Association Table Mapping*: Saves an association as a table with foreign keys to the tables that are linked by the association.
- *Dependent Mapping*: Has one class perform the database mapping for a child class.
- *Embedded Value*: Maps an object into several fields of another object's table.
- *Serialized LOB:* Saves a graph of objects by serializing them into a single large object (LOB), which it stores in a database field.

Dědičnost v OOP

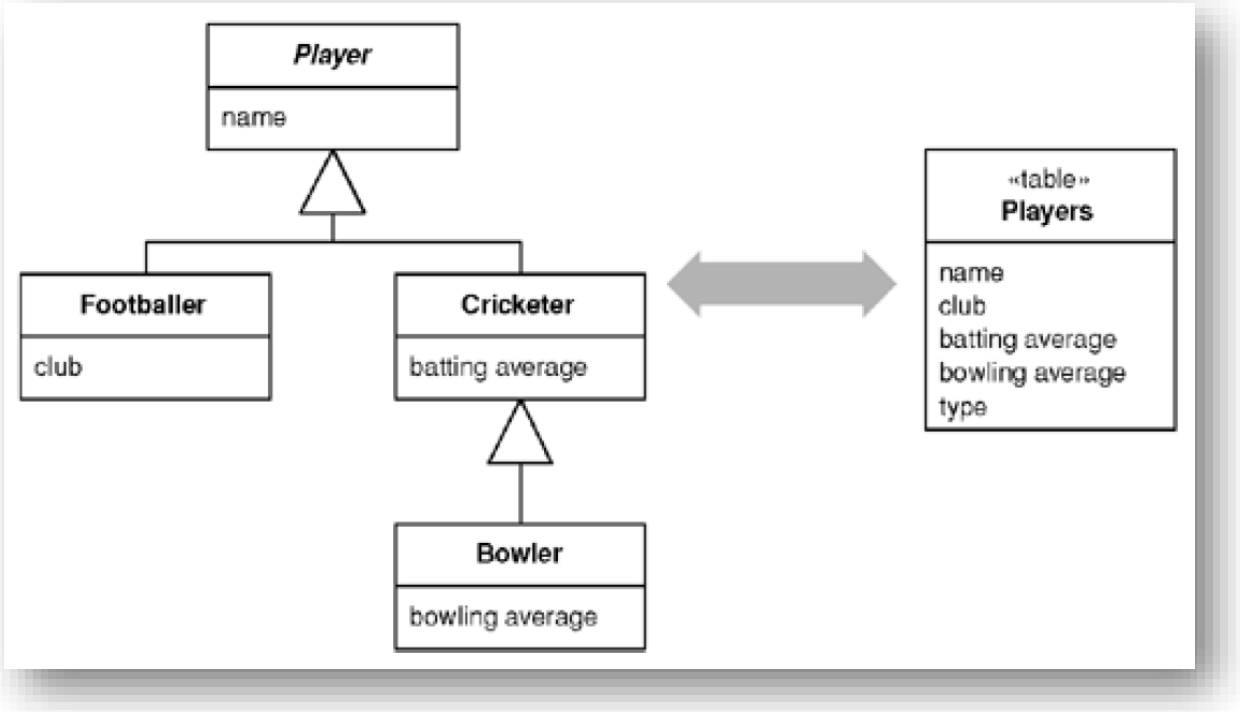
- Co?
  - Jednoduchá
  - Vícenásobná
- Proč?
  - Změna
  - Rozšíření
- Jak?
  - Data (stav)
  - Chování
- Polymorfismus a polymorfní struktury

Mapování dědičnosti

- Single Table Inheritance
- Class Table Inheritance
- Concrete Table Inheritance
- Inheritance Mappers

Single Table Inheritance

- Represents an inheritance hierarchy of classes as a single table that has columns for all the fields of the various classes.

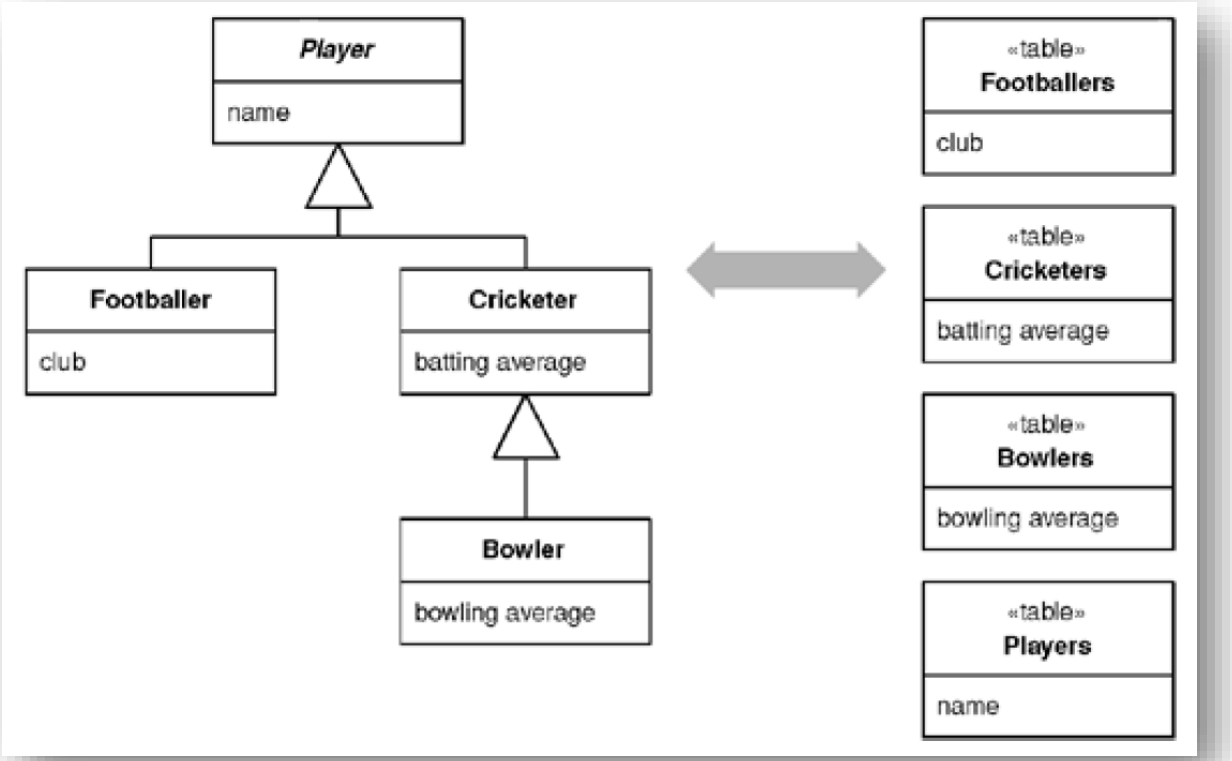


Silné a slabé stránky

- Pro:
  - V databázi je pouze jedna tabulka, o kterou se musíme starat.
  - Při získávání dat se nepoužívají žádná spojení.
  - Refaktoring, který přesunuje data nahoru nebo dolů v hierarchii tříd, nevyžaduje změnu v databázi.
- Proti:
  - Pole jsou někdy potřeba a někdy ne, což může být matoucí.pro lidi, kteří pracují přímo s tabulkami.
  - Sloupce používané pouze některými podtřídami vedou k plýtvání místem v databázi.
  - Jedna tabulka může být příliš velká, s mnoha indexy a častým zamykáním, což může ovlivnit výkon.

Class Table Inheritance

- Represents an inheritance hierarchy of classes with one table for each class.

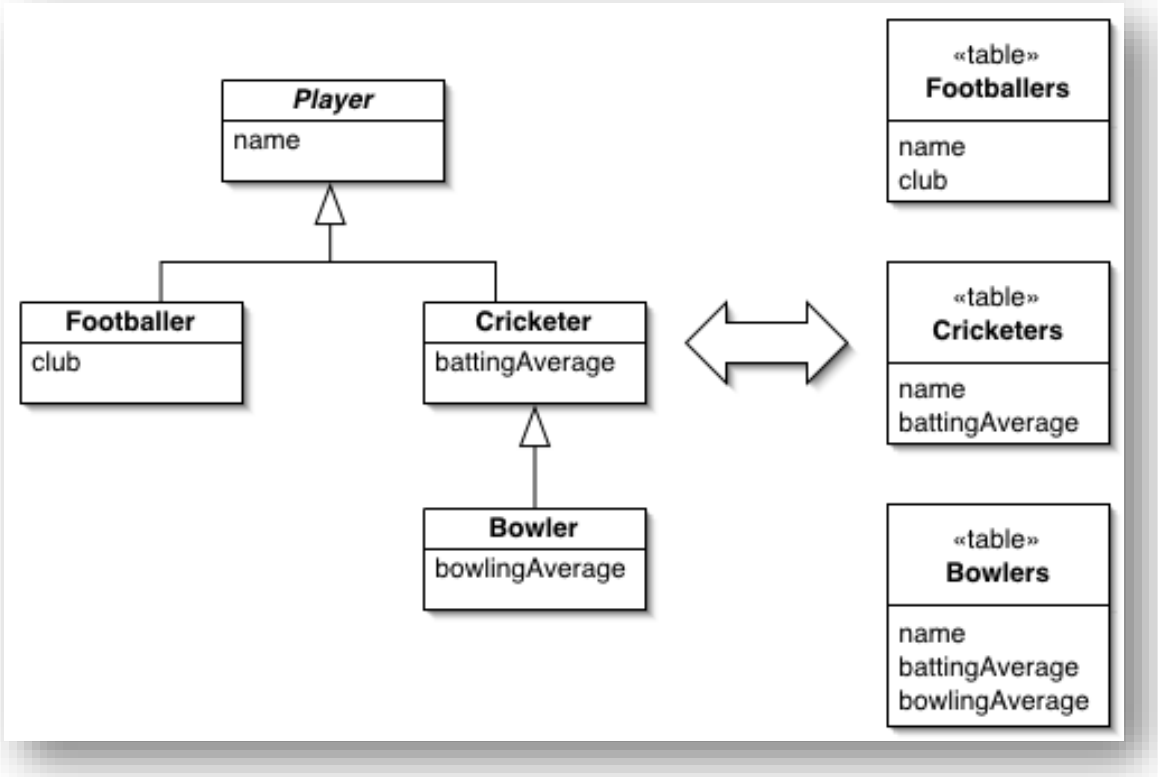


Silné a slabé stránky

- Pro:
  - Pro každý řádek jsou použité všechny sloupce, takže tabulky jsou přehlednější a neplýtváme místem.
  - Vztah mezi doménovým modelem dědičnosti a databází je přímočarý.
- Proti:
  - Pro načtení objektu je třeba se pracovat s více tabulkami, což znamená použití spojení více dotazů v paměti.
  - Jakýkoli přesun dat v dědičné hierarchii nahoru nebo dolů způsobuje změny v databázi.
  - Tabulky nadtypů se mohou stát úzkým hrdlem, protože se k nim musí často přistupovat.
  - Vysoký stupeň normalizace databáze může ztížit pochopení pro ad-hoc dotazy.

Concrete Table Inheritance

- Represents an inheritance hierarchy of classes with one table per concrete class in the hierarchy.



Silné a slabé stránky

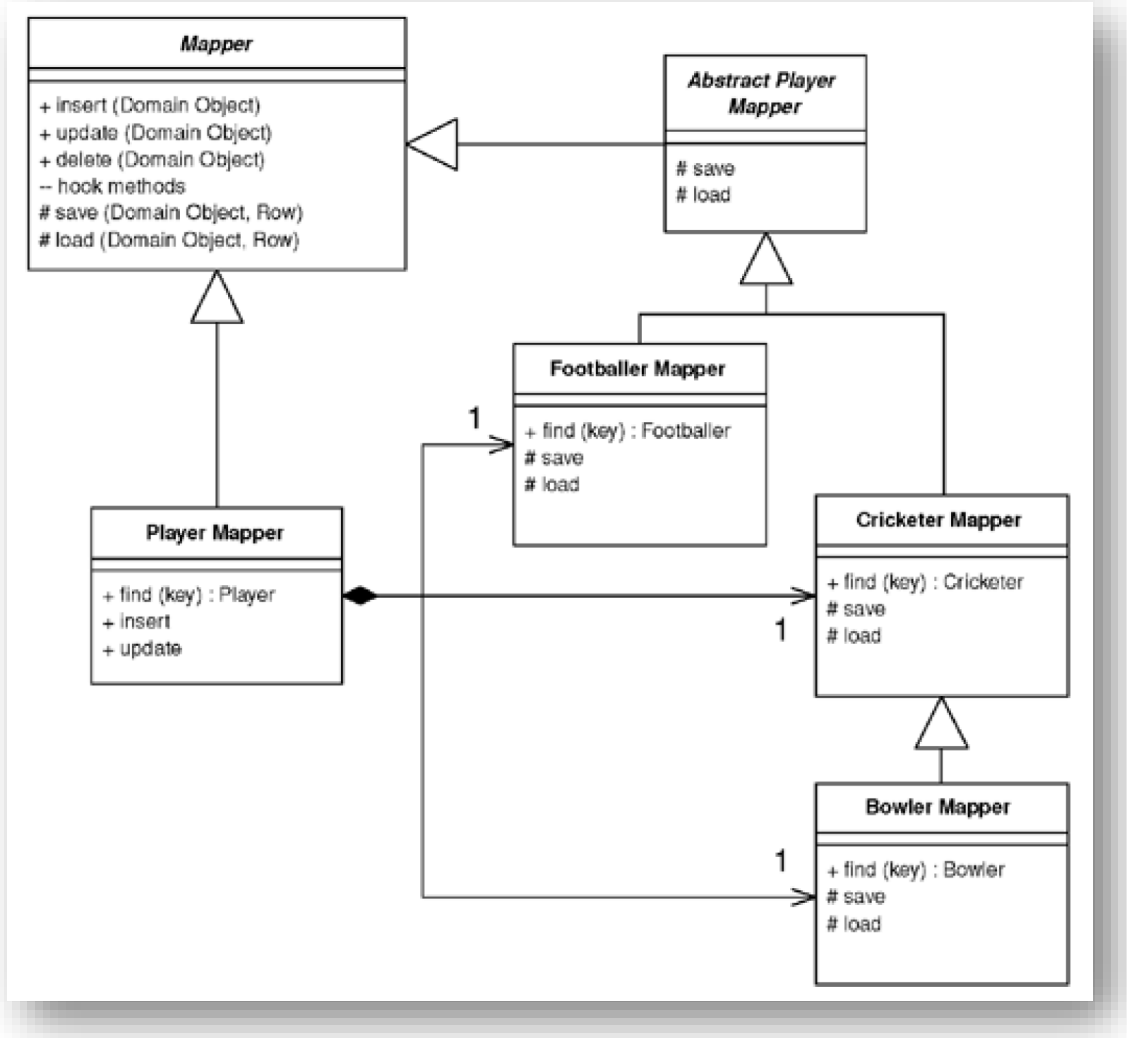
- Pro:
  - Každá tabulka je samostatná a neobsahuje žádná prázdná pole.
  - Při čtení dat z konkrétních mapperů není třeba provádět žádná spojení.
  - Ke každé tabulce se přistupuje pouze tehdy, když se přistupuje k dané třídě, což snižuje náklady na dotazování.
- Proti:
  - V databázi nelze dobře pracovat s abstraktními třídami.
  - Pokud jsou pole na doménových třídách posunuta nahoru nebo dolů v dědičné hierarchii, je nutné změnit tabulky.
  - Pokud se změní pole nadtřídy, je nutné změnit každou tabulku, která toto pole má.
  - Vyhledání nadtřídy (polymorfismus) je nutné zkontrolovat všechny tabulky, což vede k vícenásobným přístupům do databáze.

Kdy a jak je tedy použít?

- Efektivita uložení.
- Výkonnost.
- Dotazování.
- Lze je i kombinovat (např. pro různé úrovně dědičné hierarchie můžeme použít různé vzory)

Inheritance Mappers

- A general structure to organize database mappers that handle inheritance hierarchies.
- There is a need to minimize the amount of code needed to save and load the data to the database.
- Although the details of this behavior vary with the inheritance mapping scheme the general structure works the same for all of them.



Úkoly na cvičení

- Prezentace modelu domény.
- Diskuze otázek spojených se semestrálním úkolem.
- Implementace vzorů pro mapování dědičnosti.

Kontrolní otázky

1. Popište podstatu vzorů *Single* / *Class* / *Concrete Table Inheritance* a v jakých situacích je vhodné je použít.

2. Napište fragment kódu, ze kterého bude patrné, že jste použili vzor *Single* / *Class* / *Concrete Table Inheritance*.

3. Jaký je rozdíl mezi vzory *Single* / *Class* / *Concrete Table Inheritance*? Napište fragmenty kódu, ze kterých bude patrné, že jste použili tyto vzory.

K přečtení…

- Martin Fowler. *Patterns of Enterprise Application Architecture.* Addison-Wesley Professional, 2003 [278-304].