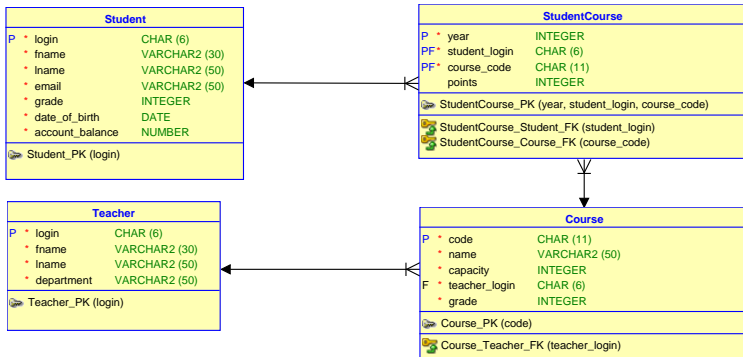


# Databázové systémy 2

db.cs@vsb.cz

Katedra informatiky  
Fakulta elektrotechniky a informatiky  
VŠB – Technická univerzita Ostrava

2021/2022





- 1 Vytvořte uloženou proceduru PPrint s parametrem p\_text, která za pomoci `dbms_output.put_line` vypíše p\_text na serverový výstup.
- 2 Vytvořte uloženou proceduru PAddStudent1 s parametry p\_login, p\_fname, p\_lname, p\_email, p\_grade a p\_dateOfBirth, která vloží nový záznam do tabulky Student.
- 3 Vytvořte proceduru PAddStudent2 pro vložení nového studenta se stejnými parametry jako PAddStudent1 kromě parametru p\_login. Login studenta bude sestaven automaticky z prvních tří písmen příjmení, ke kterým bude přidán řetězec '000'. Znaky v loginu budou vždy převedeny na malá písmena.
- 4 Vytvořte proceduru PAddStudent3. Procedura bude fungovat obdobně jako PAddStudent2. Login však bude sestaven tak, že k prvním třem písmenům z příjmení (převedených na malá písmena) budou přidány 3 číslice představující počet studentů (před vložení nového studenta) + 1.

Pozn.: Všechny napsané procedury spustíte např. příkazem EXECUTE.



- 1 Vytvořte uloženou funkci `FAddStudent1`, která bude fungovat obdobně jako procedura `PAddStudent1`. Funkce bude navíc vracet 'ok', pokud bude záznam úspěšně vložen, nebo 'error', pokud dojde k chybě (použijte část `EXCEPTION`).
- 2 Vytvořte uloženou funkci `FAddStudent2`, která bude fungovat obdobně jako procedura `PAddStudent3`. Při úspěšném vložení funkce vrátí login studenta. V případě chyby funkce vrátí 'error'.
- 3 Vytvořte uloženou funkci `FGetLogin` s parametrem `p_lname`, která vrátí login sestavený z prvních tří písmen příjmení (parametr `p_lname`) převedených na malá písmena přidáním aktuálního počtu studentů + 1 (tj. logika bude obdobná jako v `PAddStudent3`).
- 4 Vytvořte uloženou funkci `FAddStudent3` fungující obdobně jako funkce `FAddStudent2`, přičemž bude využívat funkci `FGetLogin`.

Pozn.: Funkce spusťte například výpisem pomocí procedury `PPrint`.



- 1 Vytvořte trigger `TInsertStudent`, který po vložení studenta vypíše jeho login a celé jméno. Pro vyzkoušení triggeru přidejte pomocí procedury `PAddStudent3` smyšleného studenta.
- 2 Vytvořte trigger `TDeleteStudent`, který vypíše login a celé jméno studenta před jeho odstraněním. Pro vyzkoušení triggeru odstraňte studenta vloženého v předchozím bodě.
- 3 Vytvořte trigger `TUpdateStudent`, který při aktualizaci studenta vypíše na obrazovku hodnoty atributů `fname`, `lname` a `grade` platné před a po změně. Pro vyzkoušení triggeru přerad'te studenty 'mcc676' a 'kow007' jedním příkazem `UPDATE` do (o jedna) vyššího ročníku.



- 4 Vytvořte trigger `TInsertStudent1`, který studentovi po jeho vložení do tabulky `Student` zapíše všechny předměty prvního ročníku na aktuální kalendářní rok. Pro vyzkoušení triggeru přidejte pomocí procedury `PAddStudent3` smyšleného studenta studujícího první ročník.
- 5 Vytvořte trigger `TInsertStudent2`, který před studentovi před jeho vložením do tabulky `Student` přidělí login pomocí funkce `FGetLogin()`. Pro vyzkoušení triggeru příkazem `INSERT` vložte smyšleného studenta. Po vložení se ujistěte, že byl studentovi automaticky přidělen login.
- 6 Po vyzkoušení všechny triggery vytvořené v bodech 1 až 5 odstraňte.



- 1 Vytvořte proceduru `StudentBecomeTeacher` se dvěma parametry `p_login` a `p_department`, která přesune záznam studenta s daným loginem z tabulky `Student` do tabulky `Teacher`. Pro procvičení se vyhněte použití jakýchkoli lokálních proměnných (mimo dané vstupní parametry). Procedura bude napsána tak, aby představovala jednu transakci.
- 2 Vytvořte proceduru `PStudentAssignment` s parametry `p_fname`, `p_lname` a `p_dateOfBirth`. Procedura vloží nového studenta, přičemž jeho login bude vygenerován funkcí `FGetLogin()`, e-mail bude složen z loginu přidáním '@vsb.cz' a ročník bude nastaven na 1. Procedura dále zapíše studentovi všechny předměty prvního ročníku. Procedura bude řešena jako transakce.

Pozn.: Do obou procedur přidejte výpis pomocí `PPrint`, který bude hlášením 'ok' nebo 'error' informovat, zda transakce došla v pořádku nebo zda došlo k problému.



- 1 Napište proceduru PSendEMail s parametry p\_email, p\_subject a p\_body. Procedura bude simulovat zaslání e-mailu s obsahem p\_body na adresu p\_email s předmětem p\_subject tím, že tyto parametry vypíše na serverový výstup<sup>1</sup>.
- 2 Napište trigger TSendEMail, který při přihlášení studenta ke kurzu studentovi zašle za pomoci procedury PSendEMail e-mail s předmětem 'Přihlášení ke kurzu [název kurzu]' a obsahem:  
  
*'Vážený studente [jméno a příjmení], dne [aktuální datum a čas] jste byl přihlášen ke kurzu [kód a název kurzu]. Vyučujícím kurzu je [jméno a příjmení učitele].'*

Obsah ve hranatých závorkách nahrad'te příslušnými hodnotami.

---

<sup>1</sup>Podobné procedury bývají součástí většiny relačních databázových systémů s procedurální nadstavbou.





- 1** Napište funkci `FGetStudentScore` s parametrem `p_login`, která pro studenta s daným loginem vypočte a vrátí jeho skóre. Skóre bude vypočteno jako podíl všech jeho nasbíraných bodů ku bodům, které může celkově získat. Za každý předmět může student získat maximálně 100b. Předpokládejte, že studenti žádný ze zapsaných předmětů neopakovali. Skóre studenta tedy může být v nejlepším případě rovno 1.
- 2** Napište proceduru `PCheckStudents` s parametrem `p_amount`, která všem studentům s nadprůměrným získaným počtem bodů ze všech předmětů přičte na účet částku `p_amount` a naopak všem podprůměrným studentům tuto částku odečte. Studenti, kteří budou mít po této operaci záporný stav účtu, budou vymazáni. Procedura bude napsána jako transakce.