DATABÁZOVÉ SYSTÉMY

Semestrální projekt

Patrik Florians, 194286

Oto Pokorný, 143981

David Poštulka, 170825

Garant: Ing. Jiří Kříž, Ph.D.

Vyučující: Ing. Jan Luhan, Ph.D., MSc

Obsah

1. Formulácia úlohy

Cieľom projektu je implementácia databázy za informačným systémom, ktorého úlohou je zaznamenávať informácie o vopred definovaných entitách a ich vzájomnej interakcii. Projekt sa týka len štúdia, v ktorom figurujú jednotlivé typy užívateľov. Štúdiom sa rozumie všetko od doby podania prihlášky cez platby študentov, zápočty a skúšky až po záverečnú prácu a jej hodnotenie. Kontextom do ktorého sú entity zasadené je Vysoké učení technické so svojími fakultami, ústavmi a celou hierarchiou. Ostatné orgány z vedenia školy, vedenia fakulty atď. netvoria súčasť projektu. Ostatné fyzické objekty ako napr. obsah a kapacita vyučovacích miestností taktiež netvoria súčasť projektu. Súčasťou projektu nieje ani somtný proces výučby a ani materiály s ním spojené. V podstate je bezpečné tvrdiť, že celý informačný systém slúži na administráciu štúdia v jeho generickej forme. Medzi všeobecné úkony systému patrí prezeranie, pridanie, úprava a mazanie záznamov - dát.

2. Vyhodnotenie požiadavkov na dáta

Všetky úkony sa musia uskutočňovať len v rámci adekvátnych povolení daného typu užívateľa, z čoho plynie, že povolené úkony užívateľov sa pri ich vzájomnej interakcii vždy líšia. Napríklad študent nemôže mať možnosť hodnotiť skúšku. Všetky privilégia sú v systéme definované vzhľadom na patričné roly užívateľov vyplývajúce práve z akademickej hierarchie. Zároveň existuje závislosť na interakcii entít, pretože interakciu v tomto kontexte možno definovať ako procedurálnu činnošť ktorá musí mať nejaký vstup aj výstup. Výstupom z týchto interaktívnych úkonov sú dáta, ktoré sú iné pre rôzne typy entít a taktiež nie všetky entity smú akokoľvek s týmito dátami interagovať. Všetky úkony sú obmedzené nejakými podmienkami a tie nemusia byť len typu užívateľského oprávnenia, môže ísť o podmienky definované časovým limitom a pod. Z uvedeného platí, že systém je tu poverený dozorom nad týmito úkonmi a zároveň musí zodpovedať za konzistentnosť a správnosť údajov, berúc do úvahy všetky definované limitácie a podmienky.

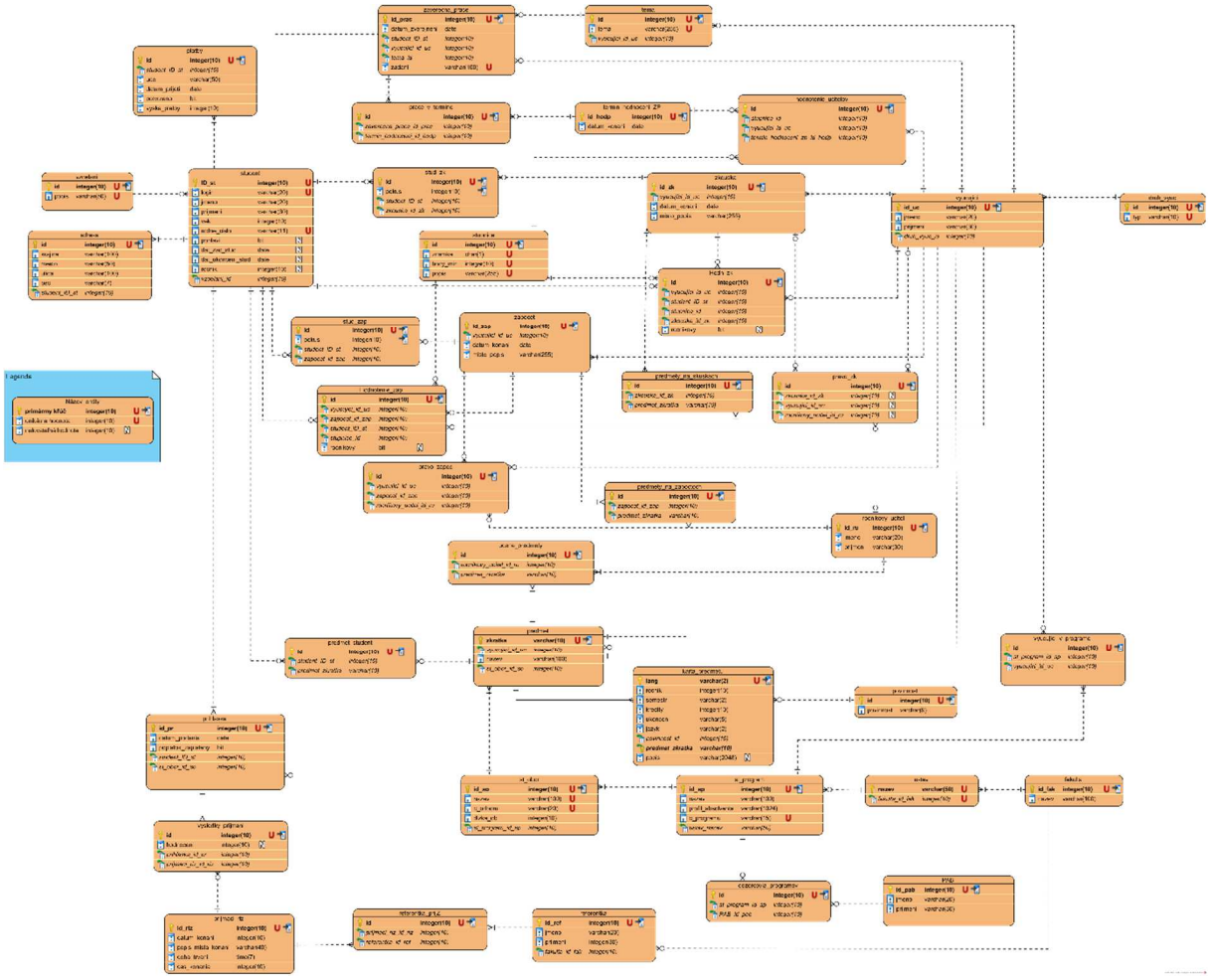
Čo sa týka samotných dát, tak tie sú pochádzjú z 3 primárnych oblastí. Prvou sú pevne definované dáta ako napríklad hodnotiaca stupnica, ktorá je v Európskej únii definovaným štandardom. Druhou sú dáta pochádzajúce priamo od užívateľov t.j. dáta ktoré užívateľ vložil do systému a tak nastalo ich vytvorenie. Treťou sú dáta, ktoré vznikli vzájomnou interakciou entít a predtým mali podobu výstupu. Tieto dáta často nadobúdajú formu číselníkových entít.

3. Určenie entít, špecifikácia atribútov a definiícia obmedzení

Pri všetkých entitách platí, že pokiaľ nieje uvedené inak, tak je atribút považovaný za Not Null. V nasledujúcich prehľadoch neuvádzame, kôli prehľadnosti všetky atribúty, ale len tie ktoré disponujú dodatočnými obmedzeniami, resp. obmedzeniami mimo normu.

3.1. Schéma logického modelu

Obrázok 1 ilustruje logický model databázy v podobe diagramu s Entito-Relačnými väzbami a príslušnými multiplicitami. Jednotlivé entity sú dopĺňané atribútmi so stanovenými dátovými typmi a ich špecifikáciou. Jednotlivá špecifikácia entít zahŕňa štandardné obmedzenia s výnimkou obmedzenia „check“. Diagram obsahuje aj malú legendu s niektorými vysvetlivkami. Obmedzenia typu check, default a iné sú detailnejšie zahrnuté v predošej kapitole.



Obrázok 1 – logick´ý model

4. Určenie a definícia pohľadov

4.1. Pohľady

Tu uvedené pohľady sú bližšie definované resp. implementované v zdrojovom súbore, ktorý je podpornou súčasťou tejto dokumentácie. Viaceré pohľady sa pritom môžu viazať na viacero entít. Napríklad informácie o prijímacom konaní sú užitočné nielen pre študenta, ale aj pre referentku tohoto príjímania keďže tvorí jeho súčasť.

Pohľady a skupiny zainterosovaných užívateľov sú uvedené nasledovne:

`create view studentPersonalInfo` - pohľad len pre študentov, kotrý zobrazí detailné informácie o profile študent, musí byť použitý v kombinácii s prodeúrou

`create view zobrazTerminyPrijimaciekAReferentky` - pohľad pre študenta a referentky

`create view zobraz_info_o_prihlaskach` – informácie o prihláškach pre referentky

`create view zobraz_platby` – používa študent aj referentka (študent v upravenej podobe v procedúre)

`create view zobraz_limitovane_info_studenti` – Limitované informácie o študentoch pre referentku

`create view zobraz_moje_prijz` – referentka, informácie o konaniach na ktorých sa má referentka zúčastniť

`create view info_o_predmetoch` – pre študentov

`create view zobraz_temy_prac` – študent, vyučujúci

`create view zobraz_info_o_pracach` – študenti a vyučujúci

`create view zobraz_terminy` – študent, vyučujúci ide o zobraznie termínov ZP

`create view zobraz_hodnotenia_zp` – študent, vyučujúci – hodnotenie udelené vyučujúcim

`create view prace_studentov` – vyučujúci

`create view zobraz_terminy_zapoctov` – študenti, vyučujúci, ročníkový učitelia

`create view zobraz_predmety_a_zapocty` – študenti, vyučujúci, ročníkový učitelia – prehľad o zápočtoch s detailami

`create view zobraz_predmety_a_skusky` - študenti, vyučujúci, ročníkový učitelia – prehľad o skúškach s detailami

`create view zobraz_prava_roc_uc_z` – vyučujúci, ročníkový učetlia

`create view zobraz_prava_roc_uc_zk` – vyučujúci, ročníkový učitelia

`create view zobraz_komplex_inf_uc` – študenti, vyučujúci, ročníkový učiteľ, referentka, PAB
`create view dozor_pab` – PAB, vyučujúci, študenti, referentka, zobrazí informácie o dohľade nad študijnými programami

4.2. Procedúry

4.2.1. Prijímacie konanie a procedúry všeobecného určenia

`create proc inputAddress` – Vloženie adresy viazanej na štodentove ID

`create proc makeLogin` – Generuje login pre študenta

`create proc regStudent` – Registruje študenta

`create proc podatPrihlasku` – Podanie prihlášky študentom

`create proc vypisPrijZ` – Vypísanie termínu prijímačiek referentkou

`create proc vykonajPlatbu` – Vykonanie platobnej operácie študentom

`create proc hodnoteniePrijmaciek` – Zadanie hodnotenia prijímacieho konania referentkou

`create proc pridatKartuPredmetu` – Vloženie karty predmetu vyučujúcim

`create proc zobrazGarantovePredmety` – Zobrazenie predmetov garantových predmetov

`create proc zobrazUcitelovePredmety` – Zobrazenie predmetov vyučujúceho, generický variant predošlej procedúry t.j. aj garant aj učiteľ

`create proc zobrazPredmRU` – Zobrazenie predmetov učených ročníkových učiteľom

`create proc pridajUcenyPredmet` – Pridanie predmetu, ktorý bude ročníkový učiteľ učiť

`create proc zobrazKartyMojichPredmetov` – Zobrazenie kariet predmetu z predmetov učených daných vyučujúcim

`create proc vypis_temu_prace` – Vypísanie témy záverečnej práce

`create proc platby_studenta` – Zobrazenie platieb študenta, nadväzuje na pohľady

`create proc vypisTerminHodnZP` – Vypísanie termínu hodnotenia ZP

4.2.2. Registrácia predmetov a delegácia právomocí

`create procedure registrujPovinnePredmety` – Zaregistrovanie predmetov definovaných ako povinné na daný semester

`create procedure registrujPredmet` – Zaregistrovanie predmetov definovaných inak ako povinné na daný semester

`create procedure delegace_prav_zk_roc_uc` – Delegácia práv hodnotiť skúšku na ročníkového učiteľa

`create procedure delegace_prav_zap_roc_uc` – Delegácia práv hodnotiť zápočet na ročníkového učiteľa

`create procedure vypis_pravomoci_ru` – Výpis práv právomocí ročníkového učiteľa

4.2.3. Záverečná práca

`create proc vyberPracu` – Výber záverečnej práce

`create proc vyberTermin` – Výber termínu hodnotenia záverečnej práce

`create proc zadaj_hodnotenie_zp` – Zadanie hodnotenia ZP

`create proc zobraz_studentove_hodnotenie` – Možnosť pozrieť si hodnotenie ZP študenta

`create proc zobrazZpStudenta` – Zobrazenie ZP študenta

`create proc getIdPraceStudenta` – Utilitná procedúra na získanie ID študentovej ZP

`create proc detailZPStudenta` – Detailné informácie o ZP študenta

`create proc getHodpOfZP` – Utilitná procedúra na získanie ID hodnotenia ZP študenta

4.2.4. Procedúry zápočtu

`create proc vypisTerminZapoctu` – Vypísanie termínu zápočtu z predmetu

`create proc registrujNaZapocet` – Registrácia na termín zápočtu

`create proc zapoctyStudenta` – Informácie o tom na ktoré zápočty a z ktorých predmetov, je študent zaregistrovaný

`create proc ohodnotZapV` – Hodnotenie zápočtu vyučujúcim

`create proc ohodnotZapRU` – Hodnotenie zápočtu ročníkovým učiteľom

`create proc vysledkyZap` – Zobrazenie výsledkov zápočtu študenta

4.2.5. Procedúry skúšky

`create proc vypisTerminSkusky` – Vypísanie termínu skúšky

`create proc registrujNaSkusku` – Registrovanie sa na termín skúšky

`create proc skuskyStudenta` – Zobrazenie informácii o skúškach študenta, zahŕňa aj informácie o predmetich, z ktorých skúšku skaldá

`create proc ohodnotZkV` – Hodnotenie skúšky vyučujúcim

`create proc ohodnotZkRu` – Hodnotenie skúšky ročníkovým učiteľom

`create proc vysledkyZk` – Informácie o výsledku skúšky

4.2.6. Ostatné

`create proc zadelDozor` – Pridelenie dozorcov za účelom dozorovania nad určitým programom

`create proc ukonci_studium` – Ukončenie štúdia s kontrolou všetkých náležitostí

4.3. Spúšte (triggers)

`create trigger platba` – Spúšť reagujúca na prebiehajúcu platbu, vykonávanú študentom

`create trigger hodZP` – Spúšť reagujúca na hodnotenie ZP práce študenta

`create trigger hodnotenie` – Spúšť reagujúca na hodnotenie prijímacieho konania

5. Fyzická implementácia a implementačná logika činností

V projekte boli použité niektoré existujúce dáta ako napríklad stupnica ECTS (1) a taktiež dáta čerpané z webových stránok VUT v Brně. Ostatné vkladané dáta boli buď vymyslené, alebo taktiež brané z vnútorných záznamov Fakulty podnikateľskej, ktoré obsahovali jednotlivé informácie.

Elementárne enity referentka, ročníkový učiteľ, vyučujúci a PAB boli vložené do databázy ručne. Dáta entít študent sa vkladajú pomocou procedúr na to určených. Medzi ostatné ručne vkladané dáta patria aj údaje o už spomenutej stupnici ECTS, formách vzdelania (2) a pod. Ide teda o entity s charakterom číselníkov.

Dáta sú využívané v procedúrach, ktorých úlohou je s nimi manipulovať, vkladať nové dáta alebo dáta mazať. Okrem niektorých špecifických procedúr ako napr. procedúra na registráciu študenta musí byť spustenie väčšiny ostatných objektov podmeinené nejakou udalosťo či dokonca postupnosťou udalostí. Tieto udalosti niesú riadené progrmovo a nepredstavujú autonómnu súčasť programu, riadi a zadáva ich programátor. Na základe toho sa dá konštatovať, že činnosti sú reprezentované procedúrami s podporou spúšťí a s využitím kurzorov. Jednotlivé súbory postupnej následnosti týchto procedúr sú uzavreté v tranzakciách, ktoré slúžia na prezentáciu týchto procedúr.

Medzi jednotlivými činnosťami je definovaná logická následnosť t.j. postupnosť realizácie jednotlivých činností prúdovo resp. závisle na sebe. Napríklad študent si musí najprv podať prihlášku, prihlásiť sa na prijimacie konanie v nejakom termíne, ktorý ale prv musí byť vypísaný, zaplatiť prihlášku, uspieť v prijímacom konaní a až potom môže študent začať študovať, resp. registrovať si predmety a zaplatiť poplatky za štúdium.

```
exec regStudent 'Jakub', 'Malostransky', 19, 19653317349, 1, 3, 'Slovensko', 'Nove
Zamky', 'Cervena 6', '851 81', @errorMsg = @chyba output;
	if (not @chyba is null)
	begin
		select @chyba;
		rollback tran reg1;
	end;
	else
	begin
		save tran reg11;
	end;
	exec podatPrihlasku 4, 1, 1, @errMsg = @chyba output;
	exec vykonajPlatbu 'prihlaska', 400, 4, '02.02.2017', @errMsg=@chyba output;
```

Uvedený úryvok kódu predstavuje práve situáciu keď sa uplatňuje logická postupnosť činností. Čo sa týka procedúry „regStudent“ tak tá ma za úlohu v sebe volať vnorenú procedúru adresa, ktorá sa stará o uloženie adresy do databázy.

```
create proc inputAddress -- vlozi adresu viazanu na studentovo id
@krajina varchar(100),
@mesto varchar(50),
@ulica varchar(100),
@psc varchar(7),
@student_id_st as int,
@errMsg as varchar(255) output
```

Z procedúry „regStudent“, sa ale volá aj procedúra „makeLogin“, ktorá generuje identifikčné prihlasovacie meno pre študenta. Tieto „podčinnosti“ sú v procedúre realizované pomocou vnorenej tranzakcie.

```
create proc makeLogin -- login je generovany studentovi
@pr as varchar(30),
@id as int,
@login as varchar(40) output
```

```
as
	set @login = 'x' + @pr + convert(varchar, floor(10*rand()))+ convert(varchar,
@id);
	set @login = convert(varchar(40), @login);
```

Konzistencia a zmysluplnosť kódu je docielená s využitím určitého množstva podmienkových resp. vetviacich príkazov, ktoré v kombinácii s podmienkami z ostatných procedúr zaisťujú plynulý prechod a realizáciu činností. Pritom, všetky procedúry aj pohľady sú pomenúvané tak aby boli samoopisné čím sa dosahuje jednoznačné premostenie medzi činnosťou a jej programovým ekvivalentom - procedúrou. V predošlom prípade by sa teda postupnosť činností dala čítať takto: zaregistruj študenta, podaj prihlášku, zaplať za prihlášku. V prípade heslovitého pomenovania sú pri všetkých objektoch, ktorých sa to týka uvedené komentáre, ktoré špecifikujú rámec pôsobnosti toho konkrétneo objektu.

V niektorých prípadoch bolo dočasne potrebné vložiť do procedúr podmienku, ktorá umožňuje preťažiť pôvodnú logiku procedúry. Ide o častý prípad najmä vtedy ak procedúra pracuje s časom a časovými veličinami resp. dátumom od ktorého je možné aby sa daná operácia vykonala, alebo nie. I keď ide o pomerne vážnu slabinu z bezpečnostného hľadiska, je táto nutná na účely testovania. Tieto skutočnosti odráža aj nesledujúci úryvok kódu:

```
create proc vykonajPlatbu -- student plati
@ucel as varchar(50),
@vyska as int,
@id_st as int,
@datum_platby varchar(40) = NULL,
@override as bit=0, -- pretazit kkontrolu casom
@errMsg as varchar(255) output
as
	if(@override = 0)
	begin
		if(@ucel like 'ZS' and convert(date, getdate())<=convert(date,
		'15.11.'+convert(varchar(4), year(convert(date, getdate()))), 104)) --
		over ci sedi semester
		begin
			...
		end;
		else if(@ucel like 'LS' and convert(date, getdate())<=convert(date,
		'15.04.'+convert(varchar(4), year(convert(date, getdate()))), 104))
		begin
			...
		end;
		else if(@ucel like 'prihlaska')
		begin
			...
		end;
		else
		begin
			set @errMsg = 'Platba v nespravnom obdobi';
		end;
	end;
	else
	begin
		if(@ucel like 'ZS') -- over ci sedi semester
		begin
			...
		end;
		else if(@ucel like 'LS')
		begin
			...
		end;
		else if(@ucel like 'prihlaska')
```

```
		begin
			...
		end;
		else
		begin
			set @errMsg = 'Platba v nespravnom obdobi';
		end;
	end;
```

Tu si možno všimnúť, že v prvej časti t.j. v podmienke if je pôvodne overovanie pomocou dátumu, no pri použití preťaženia, je možné procedúru donútiť, aby svoj algoritmus využila podľa špecificky zadaného dátumu. K predošlej procedúre je viazaná aj spúšť, ktorá má za úlohu reagovať na priebeh platby a zaistiť ostatné paralelné náležitosti, ktoré z platbou súvisia. Napríklad ak je študent prijatý a vstupuje do 1. ročníka tak mu spúšť upraví niektoré dáta v osobných informáciách ako napr. dátum začatia štúdia a číslo ročníka v ktorom má študovať.

```
else if(@uc like 'ZS' and @val = 28000 and @datPrijatia <= convert(date, '15.11.' +
convert(varchar(4), convert(int, year(convert(date, getdate())))), 104))
begin
	update platby set potvrzeno = 1 where @id_platby = id;
	if((select top 1 dat_zac_stud from student where id_st = @idSt) is null)
	begin
		if((select hodnoceni from vysledky_prijmani where prihlaska_id_pr=(select
id_pr from prihlaska where student_id_st = @idSt)) >= 50)
		begin
			update student set dat_zac_stud = convert(date, '01.10.' +
convert(varchar(4), convert(int, year(convert(date, getdate())))), 104) where
id_st=@idSt;
			update student set rocnik = 1 where id_st=@idSt;
		end;
		end;
		else
		begin
			update student set rocnik = rocnik + 1 where id_st=@idSt;
		end;
```

Predošlý úryvok kódu zobrazuje, ako pripravuje spúšť dáta v entite študent na budúce použitie. Takto je možné regulovať následnosť činností v podobe procedúr. Ak by totiž uvedené zmeny neprebehli, nemohli by nasledovať udalosti, ktoré s nimi súvisia.

V prípade platieb bolo rozhodnuté, že musí existovať rozlišovacie kritérium t.j. zmysel existencie platby, ktorý určuje premenná účel, čím sa rozlišuje, ktorá platba je na zimný semester, letný semester, prípadne či ide o poplatok za prijímacie konanie. Platba neprebehne ak je čiastka väčšia ako jedna z alternatív t.j. 28000, 10000, alebo 400. Je možné poslať menšie čiastky s rovnakým účelom, avšak ich súčet musí v stanovenom intervale dosiahnuť výšku čiastky určenú pre daný semester. V prípade platieb je ošetrené spúšťou aj absolvovanie prijimacieho konania, kde ak je platba uhradená skôr, tak je študent prijatý do štúdia a nasleduje podobná zmena ako v naposledy uvedenom úryvku kódu.

Ďalší oporný bod je registrácia predmetov, kde vždy musí byť najprv spustená procedúra registrácie povinných predmetov, ktorá študentovi registruje predmety, ktoré sú v danom ročníku povinné. V kóde je toto uskutočnené procedúrov:

```
create procedure registrujPovinnePredmety
```

Následne prebieha proces vypisovania zápočtov vyučujúcimi ako napríklad v nasledujúcom úryvku:

```
create proc vypisTerminZapoctu -- len ak vypisuje vyucujici
@id as int,
@datum as varchar(255),
```

```
@popisMiesta as varchar(255),
@zkratka as varchar(20),
@errMsg as varchar(255) output
```
Táto procedúra a jej deriváty sú použité aj pri vypisovaní prijímacieho konania, vypisovaní termínov skúšok aj záverečných prác. Ten istý prípad predstavujú aj procedúry na registráciu skúšky, alebo zápočtu.
```
create proc registrujNaZapocet
@idSt as int,
@idTermin as int,
@errMsg as varchar(255) output
```
Vo všetkých procedúrach týkajúcich sa registrácie zápočtu alebo skúšky sa kalkuluje s určitým počtom pokusov. Konkrétne ide o 2 pokusy. Zápočty a ich termíny tak ako aj skúšky sú podmienečne ohraničené časovými intervalmi, takže aj v týchto činnostiach treba počítať s existenciou využitia možnosti „podvádzať“, keď je použitý vlastnoručne vybraný dátum.

Na hodnotenie výsledkov práce slúžia procedúry typu:
```
create proc ohodnotZapV
@idUc as int,
@idZapTerm as int,
@idSt as int,
@idStup as int,
@errMsg as varchar(255) output
```
alebo
```
create proc ohodnotZkRu
@idRu as int,
@idZkTerm as int,
@idSt as int,
@idStup as int,
@errMsg as varchar(255) output
```

Kde posledné písmeno v názve indukuje, pre ktorú entitu je procedúra určená. V prípade skúšok, alebo zápočtov to môže byť vučujúci, alebo ročníkový učiteľ. Tieto skutočnosti vyplývajú zo zadania projektu. Na to aby mohol ročníkový učiteľ hodnotiť predmet je potrebné mu udeliť práva na takúto činnosť, čo je dosiahnuté procedúrami
```
create procedure delegace_prav_zk_roc_uc
```
a
```
create procedure delegace_prav_zap_roc_uc
```
obe procedúry sú tak ako predtým podobné, líšia sa len v detailoch.

Samotné hodnotenie sa vo svojom princípe odlišuje len v prípade hodnotenia záverečnej práce.
```
create proc zadaj_hodnotenie_zp
@idUc as int,
@idStupnice as int,
@idHodp as int,
@errMsg as varchar(255) output
```
V prípade tohoto hodnotenia, môže nastať situácia, že je práca ohodnotená známkou F čo indikuje, že študent neuspel pri obhajobe. Toto aktivuje spúšť, ktorá študentovy automaticky presunie termín obhajoby o 1 kalendárny rok.
```
set @znamka = (select i.stupnice_id from inserted as i);
set @idHodp = (select i.termin_hodnoceni_ZP_id_hodp from inserted as i);

if(@znamka = 6) -- ak dostal F
begin
	set @termin_kon = (select thzp.datum_konani from inserted as i,
termin_hodnoceni_ZP as thzp
	where thzp.id_hodp = i.termin_hodnoceni_ZP_id_hodp);
	set @termin_kon = dateadd(year, 1,  @termin_kon)
```

```
	update termin_hodnoceni_ZP set datum_konani = @termin_kon where id_hodp =
@idHodp;
end;
```

Samotné ukončenie štúdia má na starosti procedúra:

```
create proc ukonci_studium
@idSt as int,
@errMsg as varchar(255) output
```

, ktorej logika má na starosti kontrolu známky z výslednej práce a súčet kreditov študenta, ktorý musí dať dohromady sumu kreditov všetkých predmetov v danom štúdiu.

Ďalšia skupina procedúr sú procedúry, ktoré tvoria pseudo – pohľady. Niekedy ide o procedúry využívajúce existujúce pohľady s tým rozdielom, že určité parametre sú viazané na kontrétny argument procedúry, ktorým býva veľmi často id nejakej entity. Príklad takejto procedúry je nasledujúci:

```
create proc zobraz_studentove_hodnotenie
@idSt as int,
@errMsg as varchar(255) output
as
	if(@idSt in (select id_st from student))
	begin
		select * from zobraz_hodnotenia_zp where [ID studenta]=@idSt;
	end;
	else
	begin
		set @errMsg = 'Studentove ID nenajdene';
	end;
```

kde zobraz_hodnotenia_zp je pohľad zobrazujúci hodnotenia, všetkých prác. Keďže ide o citlivé údaje tak je výpis študentovi umožnený len na základe jeho ID.

Poslednou kategóriou procedúr sú utilitárne, alebo pomocné procedúry, ktoré sú čisto technického charakteru, a ktoré slúžia ako podporné procedúry v iných procedúrach napr:

```
create proc getHodpOfZP
@idSt as int,
@idHodp as int output,
@errMsg as varchar(255) output
as
	if(@idSt in (select id_st from student))
	begin
set @idHodp = (select id_hodp from termin_hodnoceni_zp as thzp join prace_v_termine as
pvt
	on thzp.id_hodp=pvt.termin_hodnoceni_ZP_id_hodp join zaverecna_prace as zp
	on zp.id_prac=pvt.zaverecna_prace_id_prac join student as s
	on zp.student_id_st=s.id_st
		where s.id_st=@idSt
		);
	end;
	else
	begin
		set @errMsg = 'Studentove ID nenajdene';
	end;
```

je procedúra, ktorá len vracia identifikačné číslo hodnotenia práce, čo je sám osebe nevyužiteľný fakt, ale v kombinácii s inými objektami hrá veľmi dôležitú úlohu.

6. Záver

Záverom konštatujeme, že sme vytvorili funkčnú implementáciu informačného systému z hľadiska databázového návrhu a jeho implementácie na MS SQL serveri s využitím jazyka T – SQL. Tieto tvrdenia podkladáme priloženým zdrojovým súborom a touto dokumentáciou, ktorá je doplňujúceho charakteru.

Dáta obsahnuté v súčasnej implemetácii sú po niektorých stránkach limitované svojím počtom, keďže by bolo potrebné vymýšľat značný objem týchto dát. Ďalšie obmedzenie predstavovla nedostupnosť resp. neúplnosť informácii o predmetoch zadelených do príslušných semestrov. Všetky informácie so zadania t.j. minimálne jeden konkrétny študijný program a jeho odbor boli implementované v plnom rozsahu so všetkými prislúchajúcimi náležitosťami.

7. Použité zdroje

1. **CZ, VUT.** hodnotenie podľa ECTS FIT VUT Brno. *fit.vutbr.cz.* [Online] 2017. [Dátum: 15. 12 2017.] http://www.fit.vutbr.cz/study/courses/IRP/public/obe-hodn.html.

2. **SR, MŠVVaŠ.** Národná klasifikácia vzdelania. *minedu.sk.* [Online] [Dátum: 15. 12 2017.] https://www.minedu.sk/data/files/3772.pdf.