UNIVERZITA PARDUBICE  
Fakulta elektrotechniky a informatiky

*Název projektu*

Seminární práce z předmětu Databázové systémy II

Pracovní tým:

Selecký, Kubík

V Pardubicích dne 19. 12. 2018

Obsah

[Úvod 10](#_Toc439267758)

[Zadání 11](#_Toc439267759)

[Uživatelská dokumentace 12](#_Toc439267760)

[Základní popis používané aplikace 13](#_Toc439267761)

[Instalace 13](#_Toc439267762)

[Přístupová oprávnění 13](#_Toc439267763)

[Použití aplikace 13](#_Toc439267764)

[Programová dokumentace 14](#_Toc439267765)

[Datová část 15](#_Toc439267766)

[Analýza 15](#_Toc439267767)

[Fyzický model dat 15](#_Toc439267768)

[Číselníky 15](#_Toc439267769)

[Pohledy 15](#_Toc439267770)

[Funkce 15](#_Toc439267771)

[Uložené procedury 15](#_Toc439267772)

[Spouště 15](#_Toc439267773)

[Indexy 15](#_Toc439267774)

[Sekvence 15](#_Toc439267775)

[Systémový katalog 15](#_Toc439267776)

[Aplikace 16](#_Toc439267777)

[Použité prostředí 16](#_Toc439267778)

[Řízení uživatelských účtů 16](#_Toc439267779)

[Moduly 16](#_Toc439267780)

[Formuláře 16](#_Toc439267781)

[Orientace ve zdrojovém kódu 16](#_Toc439267782)

[Závěr 17](#_Toc439267783)

Úvod

Obsahuje základní seznámení s organizací, procesy uvnitř organizace a způsob současného využívání IT.

# Zadání

Obsahuje popis prostředí, v němž bude aplikace nasazena (stručná charakteristika organizace a dosavadní využívání IT), a požadavky na nově tvořený systém (sledované údaje, vstupy a výstupy, přístupová práva, ...)

Uživatelská dokumentace

## Základní popis používané aplikace

Tato aplikace slouží pro vytváření úvazků vyučujících. Je zde tedy možné přidělovat plánovat rozvrhy jednotlivým učitelům. Tato aplikace se nezabývá tvorbou rozvrhů studentů.

## Instalace

Pro používání aplikace instalace není vyžadována. Stačí pouze spustit program s příponou .jar. Aplikaci je také možné spustit ve webovém prohlížeči, pro tuto možnost slouží soubor s příponou *.html*.

## Přístupová oprávnění

V aplikaci je umožněno vystupovat pod rolemi nepřihlášený uživatel, registrovaný uživatel a administrátor. Po spuštění aplikace je uživatel vyzván pro zadání přihlašovacích údajů. Po zadání validních údajů je uživatel přihlášen pod rolí, jež mu byla přidělena administrátorem a je uložena v databázi. Pro přístup do aplikace pod rolí nepřihlášeného uživatele je zapotřebí stisknout tlačítko *Guest* ihned po startu softwaru. Uživatel má možnost vidět svou roli na horním panelu aplikace. Každá role má rozdílná práva.

Nepřihlášený uživatel si může aplikaci pouze prohlížet a není mu dovoleno s daty jakkoliv manipulovat. Každý uživatel s touto rolí nese označení *Host,* jež je viditelné na horní liště aplikace. O tomto uživateli se neukládají do databáze žádné údaje, tudíž nemá přiděleno ani žádné heslo.

Registrovanému uživateli je umožněno vytvářet si vlastní rozvrh a obsazovat volné učebny. Dále může spravovat svůj účet jako např.: změnit si své uživatelské jméno a heslo nebo nahrát svůj profilový obrázek. Přihlašovací jméno tohoto uživatele sestává obvykle z počátečních dvou písmen *„ST“* a id uživatele. Výchozí přidělené heslo sestává ze slova *„PASSWORD“* a id uživatele. Např: login *„ST54“* a heslo „*PASSWORD54“*.

Administrátor má zpřístupněna veškerá práva a má na starosti správu aplikace. Může například přidávat, upravovat či mazat uživatele, pracoviště, učebny, předměty atp. Také má možnost vytvářet rozvrhy jednotlivým vyučujícím a spravovat složení studijních plánů. Přihlašovací údaje administrátora mohou být v podstatě totožné s údaji registrovaného uživatele, ale mohou být taktéž libovolné, např.: login *„admin“* a heslo *„admin“.*

Každá uživatel má možnost kliknout na tlačítko *Odhlásit se*, které ho vrátí zpět na přihlašovací dialog. Odkud se může přihlásit (pokud nebyl přihlášen) nebo zvolit jiný účet např. s rolí administrátora.

## Použití aplikace

Po spuštění aplikace je nutné se přihlásit do databáze, poté má uživatel možnost se přihlásit, či pokračovat jako neregistrovaný uživatel.

Hlavní část aplikace se skládá ze sedmi modulů (karet), které mají následující funkce. Karta *Vyučující* obsahuje záznamy všech uložených uživatelů s detailními informacemi. Karta *Pracoviště* nese informace týkající se všech dostupných kateder univerzity. Následující modul *Učebny* obsahuje seznam učeben. Zde je například uvedena kapacita jednotlivých učeben. Karta předměty nese výpis všech dostupných předmětů. Poté modul *Rozvrhová akce* zobrazuje rozvrh jednotlivým vyučujícím. Zde má administrátor možnost prohlížet si a plánovat rozvrhy dalším uživatelům. Na kartě *Studijní obory* jsou vyobrazeny jednotlivé studijní obory. Poslední modul *Studijní plány* nabízí možnost prohlížet si dostupné studijní plány. Je zde také možnost zobrazit si předměty obsažené v jednotlivých plánech.

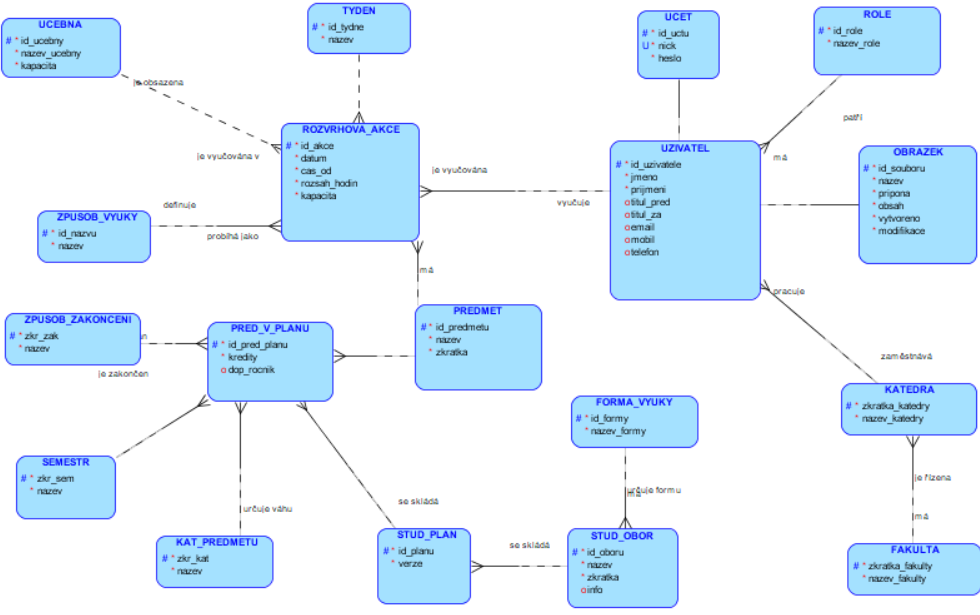
Programová dokumentace

# Datová část

V této části je blíže popsána realizace databáze od jejího návrhu až po samotnou implementaci.

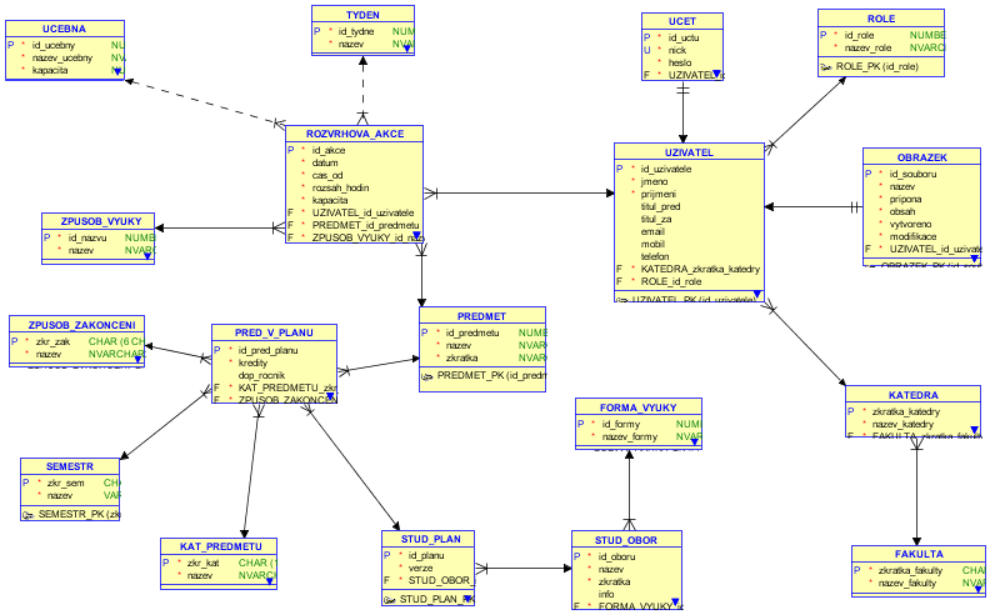
## Analýza

Entitně-vztahový model se skládá z 18 entit.



Obrázek 1: Logický datový model

## Fyzický model dat



Obrázek 2: Relační datový model

## Číselníky

Aplikace obsahuje dohromady sedm číselníků. Hodnota dat byla do jisté míry inspirována systémem Stag a následně volána v databázi následujícím způsobem.

/\*Číselník Kategorie předmětu\*/

insert into kat\_predmetu (zkr\_kat,nazev) values ('A','Povinný');

insert into kat\_predmetu (zkr\_kat,nazev) values ('B','Povinně volitelný');

insert into kat\_predmetu (zkr\_kat,nazev) values ('C','Volitelný');

alter table kat\_predmetu read only;

/\*Číselník Vyučovaný semestr\*/

insert into semestr (zkr\_sem,nazev) values ('ZS','Zimní semestr');

insert into semestr (zkr\_sem,nazev) values ('LS','Letní semestr');

alter table semestr read only;

/\*Číselník Způsob zakončení\*/

insert into zpusob\_zakonceni (zkr\_zak,nazev) values ('Zp','Zápočet');

insert into zpusob\_zakonceni (zkr\_zak,nazev) values ('Zk','Zkouška');

alter table zpusob\_zakonceni read only;

/\*Číselník Způsobu výuky\*/

insert into zpusob\_vyuky (nazev) values ('Přednáška');

insert into zpusob\_vyuky (nazev) values ('Cvičení');

insert into zpusob\_vyuky (nazev) values ('Seminář');

insert into zpusob\_vyuky (nazev) values ('Exkurze');

alter table zpusob\_vyuky read only;

/\*Číselník Forma výuky\*/

insert into forma\_vyuky (nazev\_formy) values ('Prezenční');

insert into forma\_vyuky (nazev\_formy) values ('Kombinovaná');

insert into forma\_vyuky (nazev\_formy) values ('Distanční');

alter table forma\_vyuky read only;

/\*Číselník Týdne\*/

insert into tyden (nazev) values ('Každý');

insert into tyden (nazev) values ('Sudý');

insert into tyden (nazev) values ('Lichý');

alter table tyden read only;

/\*Číselník Rolí\*/

insert into role (nazev\_role) values ('Administrátor');

insert into role (nazev\_role) values ('Registrovaný');

insert into role (nazev\_role) values ('Neregistrovaný');

alter table role read only;

## Pohledy

Aplikace sestává z šesti pohledů. Tyto pohledy jsou užívány pro naplnění jednotlivých karet aplikace.

/\*Uživatelé\*/

create or replace view uzivatel\_view as select id\_uzivatele, jmeno, prijmeni,titul\_pred,titul\_za,email,mobil,telefon,

id\_role,nazev\_role,

zkratka\_katedry,nazev\_katedry,zkratka\_fakulty,nazev\_fakulty,

id\_uctu,nick,heslo,

id\_souboru

from uzivatel

join role on id\_role = role\_id\_role

join katedra on katedra\_zkratka\_katedry = zkratka\_katedry

join fakulta fak on fakulta\_zkratka\_fakulty = fak.zkratka\_fakulty

left join ucet uc on id\_uzivatele = uc.uzivatel\_id\_uzivatele

left join obrazek obr on id\_uzivatele = obr.uzivatel\_id\_uzivatele;

/\*Pracoviště\*/

CREATE VIEW pracoviste\_view AS SELECT zkratka\_fakulty,nazev\_fakulty,zkratka\_katedry,nazev\_katedry

FROM fakulta fak JOIN katedra kat ON fak.zkratka\_fakulty=kat.fakulta\_zkratka\_fakulty;

/\*Studijní obory\*/

CREATE VIEW stud\_obory\_view AS SELECT id\_oboru, zkratka, nazev, info, id\_formy, nazev\_formy

FROM forma\_vyuky f JOIN stud\_obor s ON f.id\_formy = s.forma\_vyuky\_id\_formy;

/\*Rozvrhove akce\*/

create or replace view akce\_view as select id\_akce as , datum, cas\_od as hodina, rozsah\_hodin as doba,

rozvrhova\_akce.kapacita as mist, uzivatel\_id\_uzivatele as id\_uzivatel, titul\_pred as titul, jmeno, prijmeni ,

predmet\_id\_predmetu as id\_predmet, predmet.NAZEV as predmet,

zpusob\_vyuky\_id\_nazvu as id\_zpusobu, zpusob\_vyuky.nazev as zpusob,

ucebna\_id\_ucebny, nazev\_ucebny as ucebna,

id\_tydne, tyden.nazev as tyden

from rozvrhova\_akce

join uzivatel on id\_uzivatele = uzivatel\_id\_uzivatele

join predmet on id\_predmetu = predmet\_id\_predmetu

join zpusob\_vyuky on id\_nazvu = zpusob\_vyuky\_id\_nazvu

left join ucebna on id\_ucebny = ucebna\_id\_ucebny

left join tyden on id\_tydne = tyden\_id\_tydne;

/\*Studijní plány\*/

create or replace view plany\_view as

select id\_planu,verze,

id\_oboru,nazev,zkratka,info,

id\_formy,nazev\_formy

from stud\_plan

join stud\_obor on stud\_obor\_id\_oboru = id\_oboru

join forma\_vyuky on id\_formy = forma\_vyuky\_id\_formy;

/\*Předměty v plánu\*/

create or replace view predmety\_planu\_view as

select id\_pred\_planu as id\_predmetu\_v\_planu, kredity, dop\_rocnik as rocnik,

zkr\_kat as zkratka\_kategorie, kat\_predmetu.nazev as nazev\_kategorie,

id\_predmetu, predmet.nazev as nazev\_predmetu, predmet.zkratka,

zkr\_zak, zpusob\_zakonceni.nazev as nazev\_zpusobu\_zak,

zkr\_sem as zkratka\_semestru, semestr.nazev as semestr,

id\_planu,verze,

id\_oboru, stud\_obor.nazev as nazev\_oboru,stud\_obor.zkratka as zkratka\_oboru,info,

id\_formy,nazev\_formy as forma

from pred\_v\_planu

join kat\_predmetu on zkr\_kat = pred\_v\_planu.kat\_predmetu\_zkr\_kat

join predmet on id\_predmetu = pred\_v\_planu.predmet\_id\_predmetu

join zpusob\_zakonceni on zkr\_zak = pred\_v\_planu.zpusob\_zakonceni\_zkr\_zak

join semestr on zkr\_sem = pred\_v\_planu.semestr\_zkr\_sem

join stud\_plan on id\_planu = pred\_v\_planu.stud\_plan\_id\_planu

join stud\_obor on id\_oboru = stud\_plan.stud\_obor\_id\_oboru

join forma\_vyuky on id\_formy = stud\_obor.forma\_vyuky\_id\_formy;

## Funkce

V aplikaci jsou užity dvě funkce. A to funkce pro získání ID uživatele z přihlašovacích údajů a funkce pro ověření unikátnosti uživatelského jména.

create or replace FUNCTION vratIdUzivatele (p\_nick UCET.NICK%TYPE, p\_heslo UCET.HESLO%TYPE)

RETURN UCET.UZIVATEL\_ID\_UZIVATELE%TYPE IS

v\_id\_uziv UCET.UZIVATEL\_ID\_UZIVATELE%TYPE;

BEGIN

SELECT UZIVATEL\_ID\_UZIVATELE INTO v\_id\_uziv

FROM UCET

WHERE NICK = p\_nick AND HESLO = p\_heslo;

RETURN v\_id\_uziv;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

raise\_application\_error(-20001, 'Neexistující jméno nebo heslo.');

END;

create or replace FUNCTION jeJmenoUnikatni (p\_nick UCET.NICK%TYPE)

RETURN BOOLEAN IS

v\_jeUnikatni BOOLEAN := FALSE;

v\_nick UCET.NICK%TYPE;

BEGIN

SELECT NICK INTO v\_nick

FROM UCET

WHERE NICK = p\_nick;

RETURN v\_jeUnikatni;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

v\_jeUnikatni := TRUE;

RETURN v\_jeUnikatni;

END;

## Uložené procedury

V projektu jsou procedury pro vkládání, mazání a úpravu dat v aplikaci. Procedury byly použity pro manipulaci dat tabulek *STUD\_OBOR, STUD\_PLAN, UZIVATEL, UCET, OBRAZEK, PŘEDMĚT, UCEBNA, KATEDRA, PRED\_V\_PLANU, ROZVRHOVA\_AKCE*. Z důvodu rozsáhlosti kódu lze pro detail jednotlivých procedur nahlídnout do souboru *procedury.sql*.

## Spouště

Uveďte jaké spouště Váš projekt používá včetně SQL všech použitých triggers.

/\*Trigger Způsobu výuky\*/

CREATE OR REPLACE TRIGGER TRIG\_ZPUSOB\_VYUKY

BEFORE INSERT ON ZPUSOB\_VYUKY

FOR EACH ROW

WHEN (new.ID\_NAZVU IS NULL)

BEGIN

SELECT ZPUSOB\_VYUKY\_SEQ.NEXTVAL

INTO :new.ID\_NAZVU

FROM dual;

END;

/\*Trigger Forma výuky\*/

CREATE OR REPLACE TRIGGER TRIG\_forma\_vyuky

BEFORE INSERT ON forma\_vyuky

FOR EACH ROW

WHEN (new.id\_formy IS NULL)

BEGIN

SELECT forma\_vyuky\_SEQ.NEXTVAL

INTO :new.id\_formy

FROM dual;

END;

/

/\*Trigger Týdne\*/

CREATE OR REPLACE TRIGGER TRIG\_TYDEN

BEFORE INSERT ON TYDEN

FOR EACH ROW

WHEN (new.ID\_TYDNE IS NULL)

BEGIN

SELECT TYDEN\_SEQ.NEXTVAL

INTO :new.ID\_TYDNE

FROM dual;

END;

/

/\* Trigger Rolí\*/

CREATE OR REPLACE TRIGGER TRIG\_ROLE

BEFORE INSERT ON ROLE

FOR EACH ROW

WHEN (new.ID\_ROLE IS NULL)

BEGIN

SELECT ROLE\_SEQ.NEXTVAL

INTO :new.ID\_ROLE

FROM dual;

END;

/

/\* Trigger predmetu\*/

CREATE OR REPLACE TRIGGER TRIG\_PREDMET

BEFORE INSERT ON PREDMET

FOR EACH ROW

WHEN (new.id\_predmetu IS NULL)

BEGIN

SELECT predmet\_SEQ.NEXTVAL

INTO :new.id\_predmetu

FROM dual;

END;

/

/\*Trigger Uceben\*/

CREATE OR REPLACE TRIGGER TRIG\_UCEBNA

BEFORE INSERT ON UCEBNA

FOR EACH ROW

WHEN (new.id\_ucebny IS NULL)

BEGIN

SELECT ucebna\_SEQ.NEXTVAL

INTO :new.id\_ucebny

FROM dual;

END;

/

/\*Trigger Uctu\*/

CREATE OR REPLACE TRIGGER TRIG\_UCET

BEFORE INSERT ON UCET

FOR EACH ROW

WHEN (new.id\_uctu IS NULL)

BEGIN

SELECT UCET\_SEQ.NEXTVAL

INTO :new.id\_uctu

FROM dual;

END;

/

/\*Trigger Uzivatelu\*/

CREATE OR REPLACE TRIGGER TRIG\_UZIVATEL

BEFORE INSERT ON UZIVATEL

FOR EACH ROW

WHEN (new.id\_uzivatele IS NULL)

BEGIN

SELECT UZIVATEL\_SEQ.NEXTVAL

INTO :new.id\_uzivatele

FROM dual;

END;

/

/\*Trigger oboru\*/

CREATE OR REPLACE TRIGGER TRIG\_stud\_obor

BEFORE INSERT ON stud\_obor

FOR EACH ROW

WHEN (new.id\_oboru IS NULL)

BEGIN

SELECT stud\_obor\_SEQ.NEXTVAL

INTO :new.id\_oboru

FROM dual;

END;

/\*Trigger Pred\_V\_Planu\*/

CREATE OR REPLACE TRIGGER TRIG\_pred\_v\_planu

BEFORE INSERT ON pred\_v\_planu

FOR EACH ROW

WHEN (new.id\_pred\_planu IS NULL)

BEGIN

SELECT pred\_v\_planu\_SEQ.NEXTVAL

INTO :new.id\_pred\_planu

FROM dual;

END;

/

/\*Trigger Rozvrhova\_Akce\*/

CREATE OR REPLACE TRIGGER TRIG\_rozvrhova\_akce

BEFORE INSERT ON rozvrhova\_akce

FOR EACH ROW

WHEN (new.id\_akce IS NULL)

BEGIN

SELECT rozvrhova\_akce\_SEQ.NEXTVAL

INTO :new.id\_akce

FROM dual;

END;

/

/\*Trigger STUD\_PLAN\*/

CREATE OR REPLACE TRIGGER TRIG\_STUD\_PLAN

BEFORE INSERT ON STUD\_PLAN

FOR EACH ROW

WHEN (new.ID\_PLANU IS NULL)

BEGIN

SELECT STUD\_PLAN\_SEQ.NEXTVAL

INTO :new.ID\_PLANU

FROM dual;

END;

/

/\*Trigger Obrazek\*/

CREATE OR REPLACE TRIGGER TRIG\_obrazek

BEFORE INSERT ON obrazek

FOR EACH ROW

WHEN (new.id\_souboru IS NULL)

BEGIN

SELECT obrazek\_SEQ .NEXTVAL

INTO :new.id\_souboru

FROM dual;

END;

/

## Indexy

Uveďte jaké typy indexů používáte, vypište všechny indexy na neklíčových sloupcích.

## Sekvence

Sekvence používáme pro generovaní id jednotlivých záznamů v tabulkách. V aplikaci se jich vyskytuje

/\*Sekvence Způsobu výuky\*/

CREATE SEQUENCE ZPUSOB\_VYUKY\_SEQ

START WITH 1

INCREMENT BY 1;

/

/\*Sekvence Forma výuky\*/

CREATE SEQUENCE forma\_vyuky\_SEQ

START WITH 1

INCREMENT BY 1;

/

/\*Sekvence Týdne\*/

CREATE SEQUENCE TYDEN\_SEQ

START WITH 1

INCREMENT BY 1;

/

/\*Sekvence Rolí\*/

CREATE SEQUENCE ROLE\_SEQ

START WITH 1

INCREMENT BY 1;

/

/\*Sekvence predmetu\*/

CREATE SEQUENCE PREDMET\_SEQ

START WITH 1

INCREMENT BY 1;

/

/\*Sekvence Uceben\*/

CREATE SEQUENCE UCEBNA\_SEQ

START WITH 1

INCREMENT BY 1;

/

/\*Sekvence Uzivatelu\*/

CREATE SEQUENCE UZIVATEL\_SEQ

START WITH 1

INCREMENT BY 1;

/

/\*Sekvence Uctu\*/

CREATE SEQUENCE UCET\_SEQ

START WITH 1

INCREMENT BY 1;

/

/\*Sekvence oboru\*/

CREATE SEQUENCE stud\_obor\_SEQ

START WITH 1

INCREMENT BY 1;

/

/\*Sekvence Pred\_V\_Planu\*/

CREATE SEQUENCE pred\_v\_planu\_SEQ

START WITH 1

INCREMENT BY 1;

/

/\*Sekvence Rozvrhova\_Akce\*/

CREATE SEQUENCE rozvrhova\_akce\_SEQ

START WITH 1

INCREMENT BY 1;

/

/\*Sekvence STUD\_PLAN\*/

CREATE SEQUENCE STUD\_PLAN\_SEQ

START WITH 1

INCREMENT BY 1;

/

/\*Sekvence Obrazek\*/

CREATE SEQUENCE obrazek\_SEQ

START WITH 1

INCREMENT BY 1;

/

## Systémový katalog

Vypište min. 10 vlastností Vámi vybraných databázových objektů.

# Aplikace

Následují část detailně popisuje strukturu aplikace a další náležitosti spojené s její tvorbou.

## Použité prostředí

Pro vytvoření aplikace byl zvolen jazyk Java v kombinaci s vývojovým prostředím *Netbeans*. Dále byl, zejména pro vytvoření vzhledu aplikace, využit software *Scene Builder* od společnosti *Gluon.*

## Řízení uživatelských účtů

Do aplikace lze přistupovat na základě rolí *nepřihlášený uživatel*, *registrovaný uživatel* a *administrátor*. Tyto role, kromě role *nepřihlášeného uživatele*, jsou zaznamenané u každého uživatele v databázi.

## Moduly

Aplikace se dělí na jednotlivé karty. Jsou jimi karty *Vyučující*, *Pracoviště*, *Učebny*, *Předměty*, *Rozvrhová akce*, *Studijní Obory* a *Studijní Plány*.

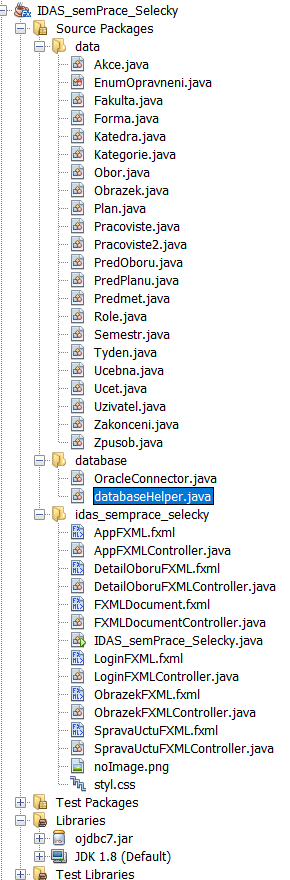
## Formuláře

Aplikace využívá několik formulářů. Každý z nich je unikátní pro danou kartu. Nejčastěji využívanými komponentami pro jejich tvorbu je *TextField*, *ComboBox* a *Spinner*. Každá z nich je opatřena odpovídajícími popisky. Dále každý formulář obsahuje tlačítka, jež zajišťují přidávání, úpravu a mazání záznamů. Pro přidání záznamu je potřeba vyplnit požadovaná pole a stisknout odpovídající tlačítko. Pro úpravu a smazání záznamu je zapotřebí nějaký záznam nejdříve vybrat. Pokud bychom chtěli ihned po vybrání záznamu přidat nějaké pole, je nutné vyplněná pole vyčistit kliknutím na tlačítko s odpovídajícím názvem. Formuláře se od sebe většinou liší počtem a typem využitých komponent. Některé formuláře mohou nabízet i další funkce. Například formulář na kartě vyučující nabízí správu účtu či obrázku uživatele.

## Orientace ve zdrojovém kódu

Projekt je rozdělen do tří balíčků, a to konkrétně do *data*, *database* a *idas\_semprace\_selecky*. První balíček obsahuje třídy, které jsou ve většině případů obdobou tabulek fyzického modelu databáze a atributy těchto tříd poté představují jednotlivé sloupce. Každá z těchto tříd kromě atributů také obsahuje metody *get*, *set* a *toString* pro manipulaci se zmíněnými atributy.

Balíček „*database*“ obsahuje třídy, jež jsou nezbytné pro komunikaci s databází. Třída *OracleConnector* slouží k navázání spojení s databází za využití metody *setUpConnection*. Třída *DatabaseHelper* zajišťuje získávání dat z databáze, ale také jejich přidávání, úpravu a mazání. Metody začínající na *dejKartu* obstarávají získávání dat, jež poté slouží k naplnění tabulek na jednotlivých kartách. Tato akce je nejčastěji prováděna za využití pohledů. Funkce, jež slouží pro přidání jednotlivých záznamů na základě požadavku uživatele začínají slovem *insert*. Funkce pro úpravu a mazání začínají na *update* a *delete*. Pro všechny tyto typy manipulace s daty jsou využity procedury.

Poslední balíček obsahuje třídy, jež slouží k vytvoření hlavní logiky programu, stejně jako ztvárnění vzhledu samotné aplikace. Pro úpravu vzhledu slouží soubory s příponou *.fxml*, jež byly vytvořeny za pomoci softwaru *Scene Builder*. K těmto souborů se vztahují třídy s názvem, který je takřka totožný, pouze na konci obsahují slovo *Controller*. Tyto třídy dále pracují s komponentami *.fxml* souborů a dodávají programu potřebnou logiku. Na začátku každé z těchto tříd dochází k deklaraci zmiňovaných komponent. Dále následuje metoda *initialize()*, jež má za úkol inicializovat tyto komponenty a mimo jiné také naplnění formulářů při kliknutí na záznam v tabulce. Poté tyto třídy taktéž obsahují metody pro aktualizaci, vkládání, úpravu a mazaní dat. Pro tento typ akcí jsou volány metody z již zmiňované třídy *DatabaseHelper*. Nejvýznamnější z těchto tříd s přívlastkem *Controller* je *AppFXMLController*, jež obsahuje hlavní část programu se všemi moduly aplikace. Dále je v tomto balíčku obsažen výchozí obrázek uživatele a kaskádový styl, který blíže specifikuje vzhled pozadí a písma, jako je například velikost nebo barva fontu.

Obrázek 3: Struktura projektu

Závěr

V závěru pochvalně i kriticky zhodnoťte realizovanou databázovou aplikaci a uveďte, jak by se aplikace mohla v budoucnu využívat.

**Přílohy**

1. Skripty SQL
2. Zdrojové kódy aplikace, grafika, apod. vše v jednom zabaleném ZIPU!
3. Samostatně spustitelná aplikace.
4. Ostatní (libovolné soubory doplňující Vaši tvorbu databázové aplikace či nutné nestandardní doplňky nutné pro spuštění Vaší aplikaci)